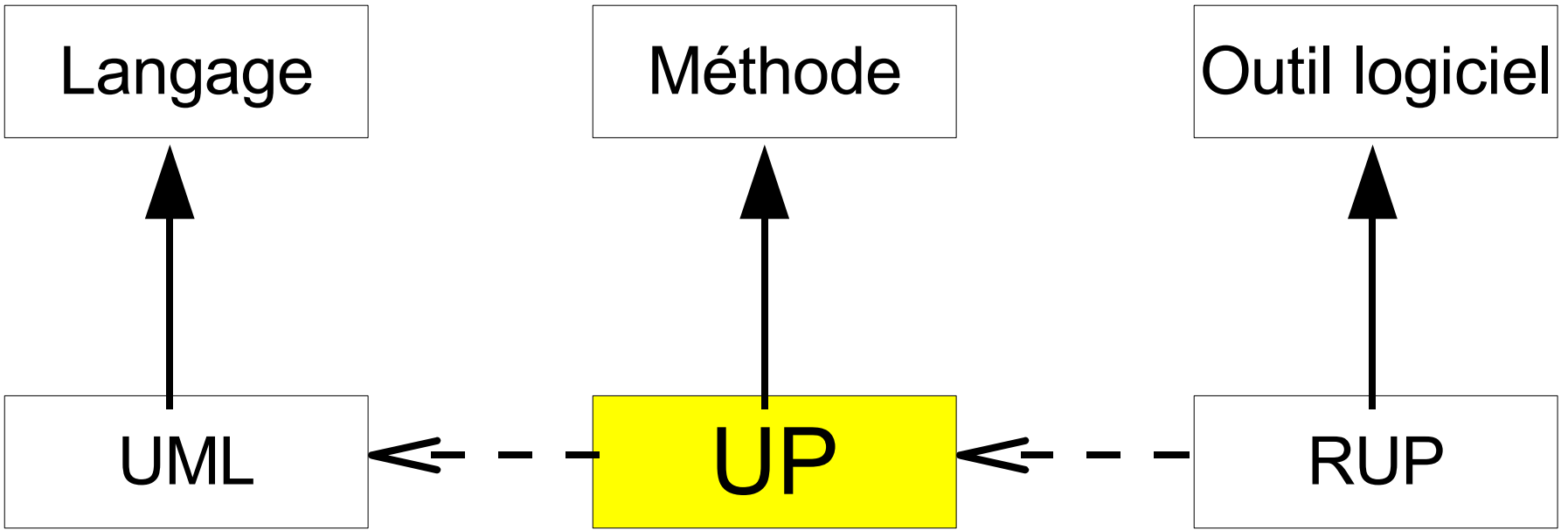
UNIFIED

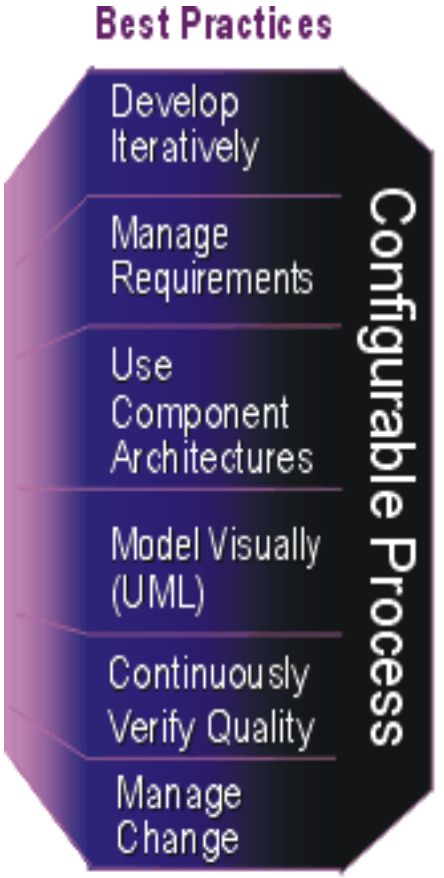
PROCESS

Positionnement de UP

Unified Process



6 BONNES PRATIQUES

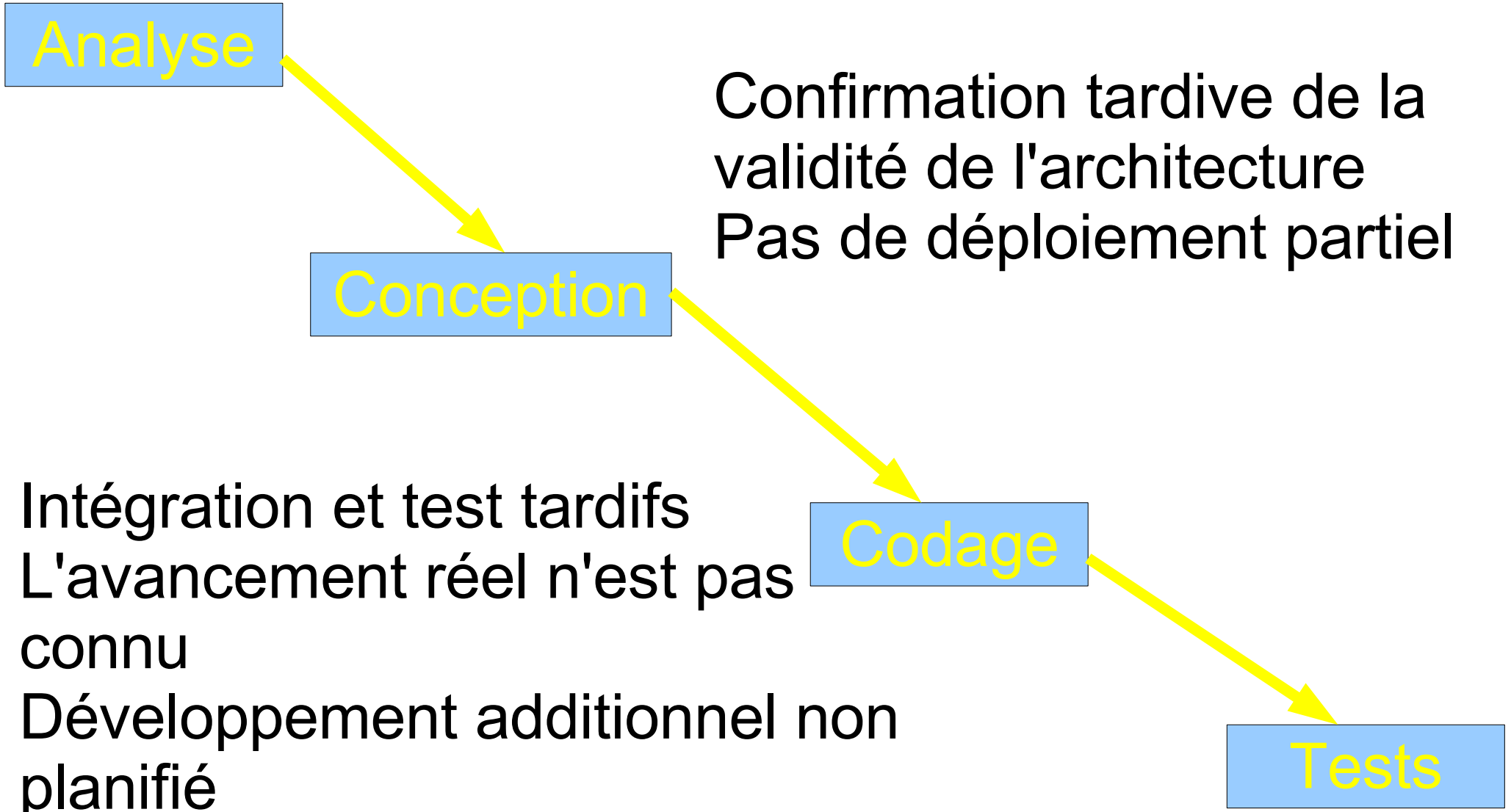


- développement itératif
- gestion des exigences
- architecture basée sur des composants
- modélisation visuelle
- vérification continuelle de la qualité
- gestion des modifications

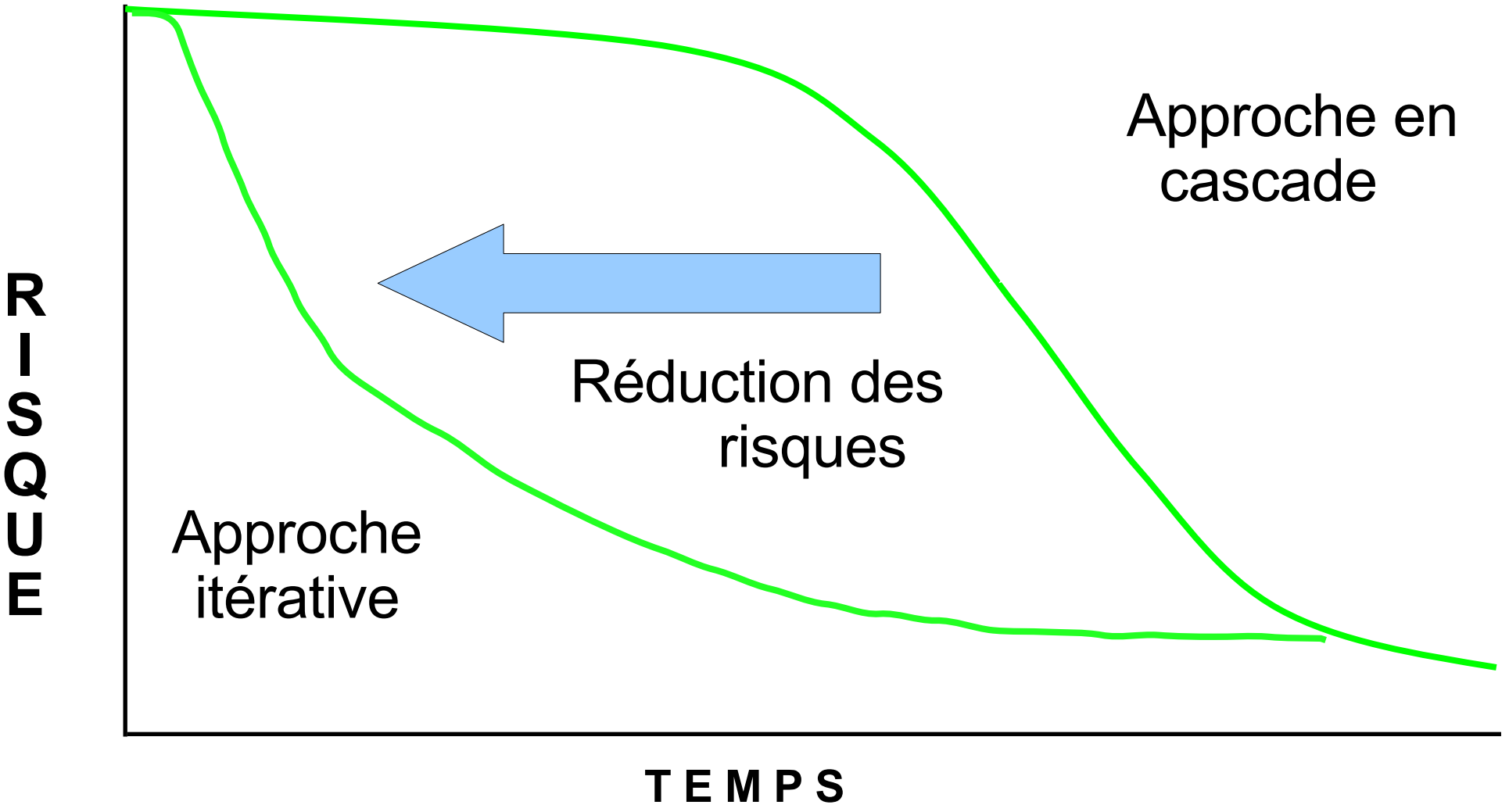
DEVELOPPEMENT ITERATIF

- Technique utilisée pour construire un système, sous forme d'une série de version, de plus en plus complet
- Chaque version est développée dans un intervalle de temps appelé itération
- Chaque itération effectue la définition, l'analyse, la conception, l'implémentation et le test d'un ensemble d'exigences

CRITIQUE DU DEVELOPPEMENT EN CASCADE



PROFIL DES RISQUES



GESTION DES EXIGENCES

- Implique l'expression des demandes par les parties prenantes
- Exigences fonctionnelles et non fonctionnelles
- Il faut non seulement les accepter mais aussi les gérer (priorité, tracabilité, test)

ARCHITECTURE A BASE DE COMPOSANTS

- Architecture évolutive et robuste
- La difficulté est d'anticiper l'évolution du domaine et des technologies
- Un composant est une partie, remplaçable d'un système, qui assure une fonctionnalité précise

MODELISATION VISUELLE

- Capturer la structure et le comportement du système
- Faciliter le passage
  conception <--> implémentation
- Masquer ou exposer des détails en fonction du contexte
- Communiquer sans ambiguïté

VERIFICATION DE LA QUALITE

- Etablir des mesures et des critères de test
- Utiliser une méthodologie claire
- Automatiser au maximum les tests
- Vérifier toutes les dimensions de la qualité (fonctionnalités, fiabilité, performance ...)
- Tester à chaque itération

GESTION DES CHANGEMENTS

- Contrôler quand et comment les changements sont introduits dans le projet
- Etablir un espace de travail sécurisé pour chaque développeur
- Automatiser l'intégration

Le processus unifié en bref

C'est un processus de développement logiciel, regroupant les activités à mener pour transformer les besoins des utilisateurs en système logiciel.

- Le processus unifié est à base de composants
- Le processus unifié utilise le langage UML

- Le processus unifié est piloté par les cas d'utilisation
- Le processus unifié est centré sur l'architecture
- Le processus unifié est itératif et incrémental

Piloté par les cas d'utilisation

Un cas d'utilisation est une fonctionnalité du système produisant un résultat satisfaisant pour l'utilisateur.

Le cas d'utilisation guide l'ensemble du processus de développement

Centré sur l'architecture “logicielle”

L'architecture est issue des cas d'utilisation et influencée par de nombreux facteurs :

- La plate-forme matérielle
- Le système d'exploitation
- Le SGBD
- Le réseau
- etc..

L'architecture est concue de façon à permettre l'évolution du système.

Itératif et incrémental

Les itérations désignent des étapes de l'enchaînement d'activités.

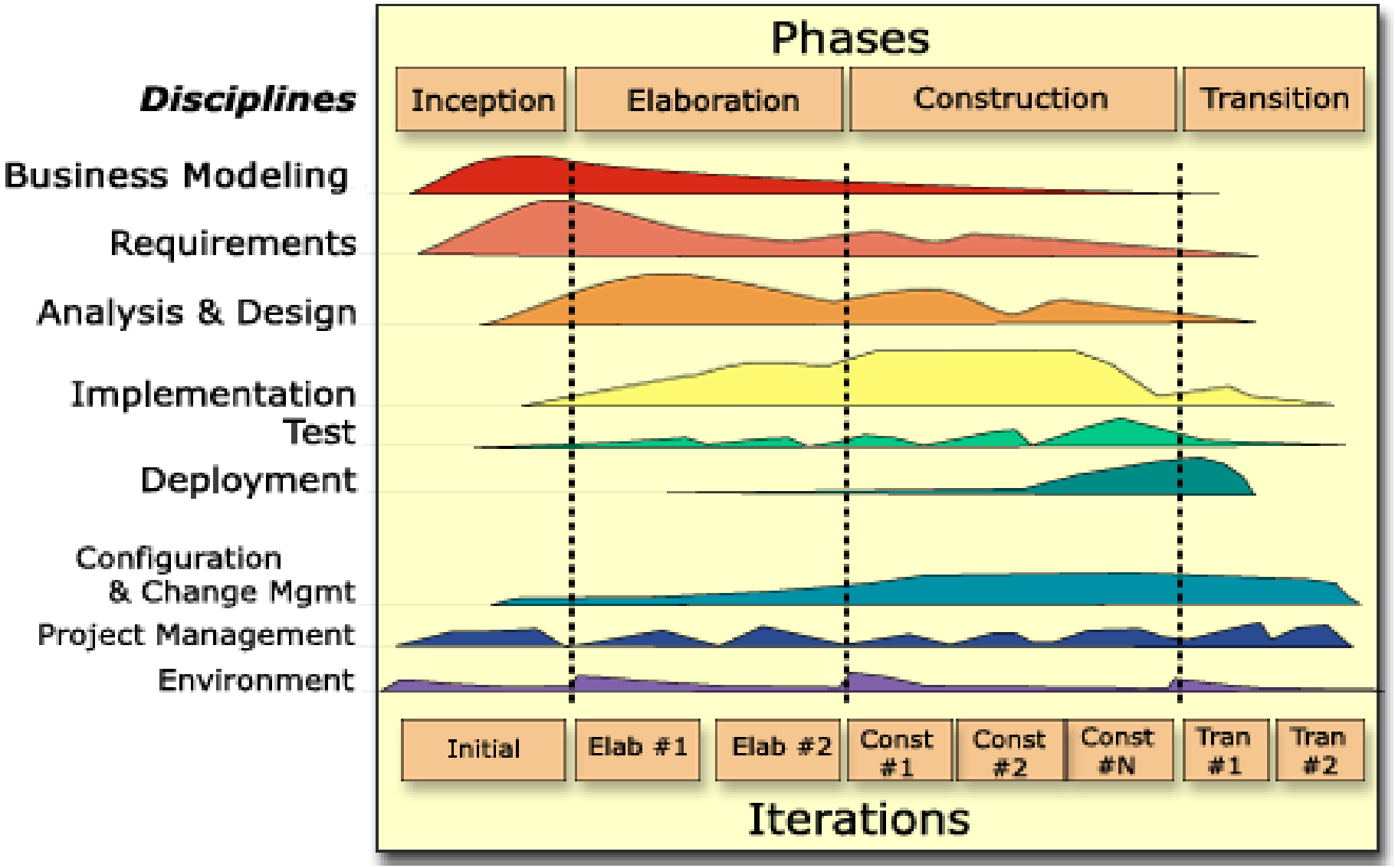
Les incréments correspondent à des stades de développement du système.

Les itérations doivent être planifiées.

Une itération

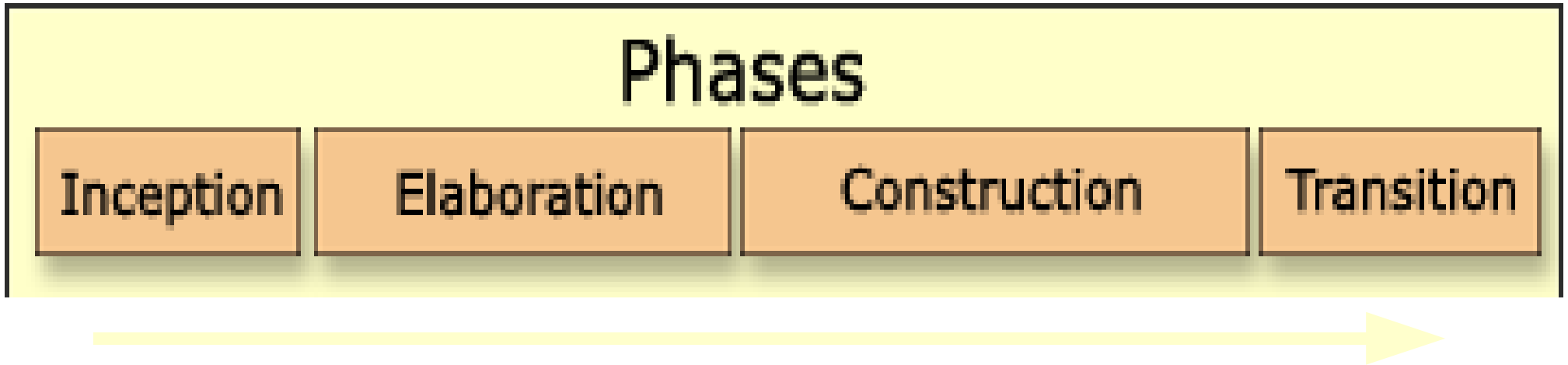
- prend en compte un certain nombre de cas d'utilisation.
- traite en priorité les risques majeurs;
- exploite les artefacts dans l'état ou les a laissés l'itération précédente

2 DIMENSIONS



4 PHASES

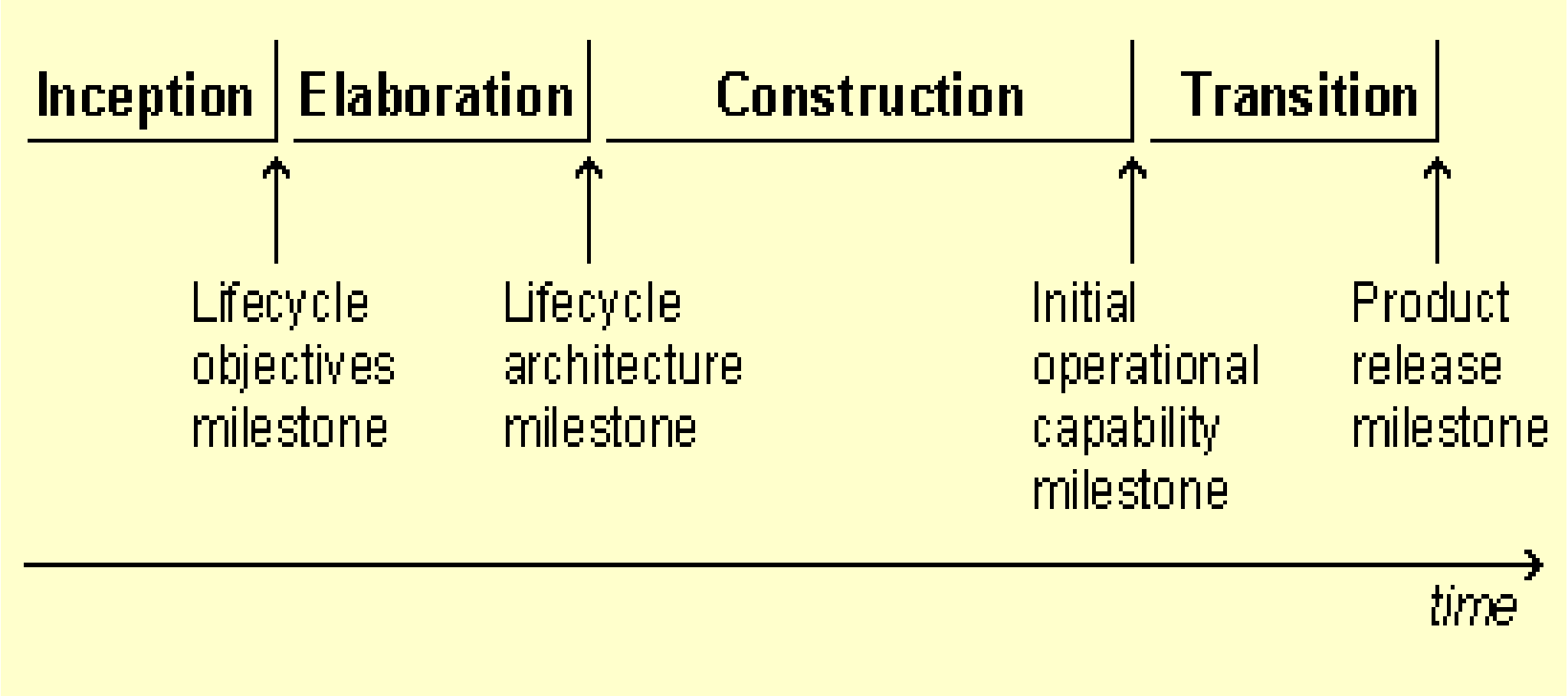
Aspect dynamique d'UP



Chaque phase :

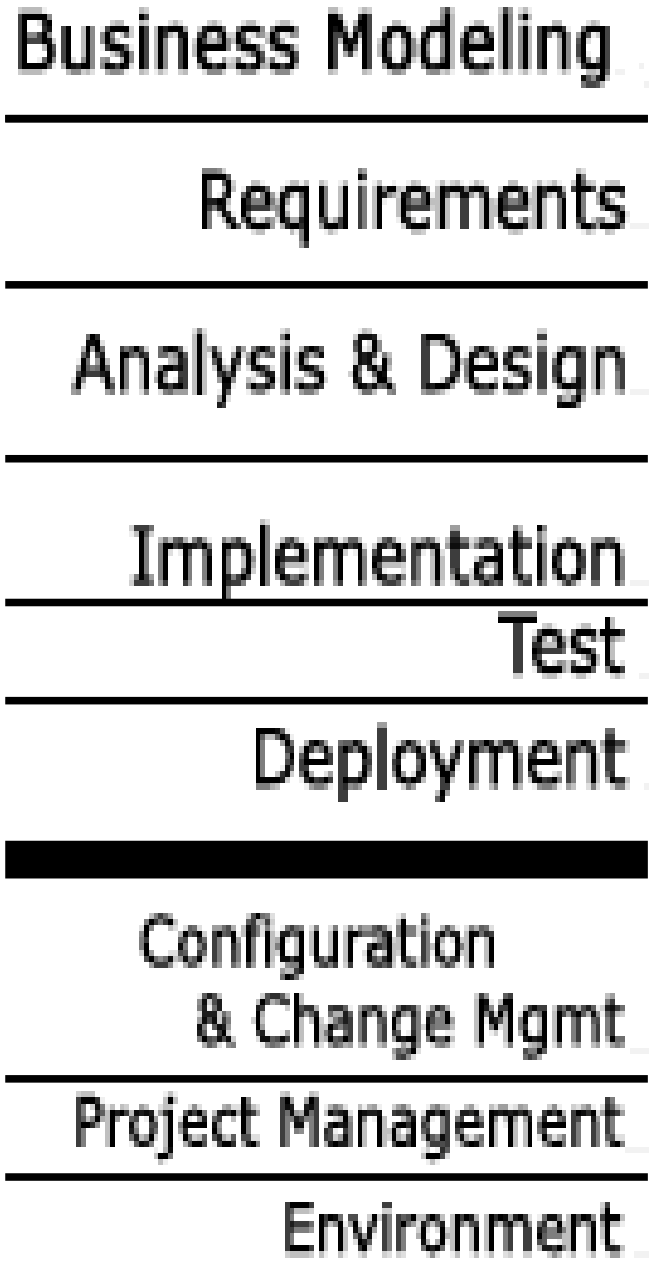
- est découpée en itérations
- a des objectifs
- se termine par un jalon

Les Phases et leur Jalon



Aspect statique

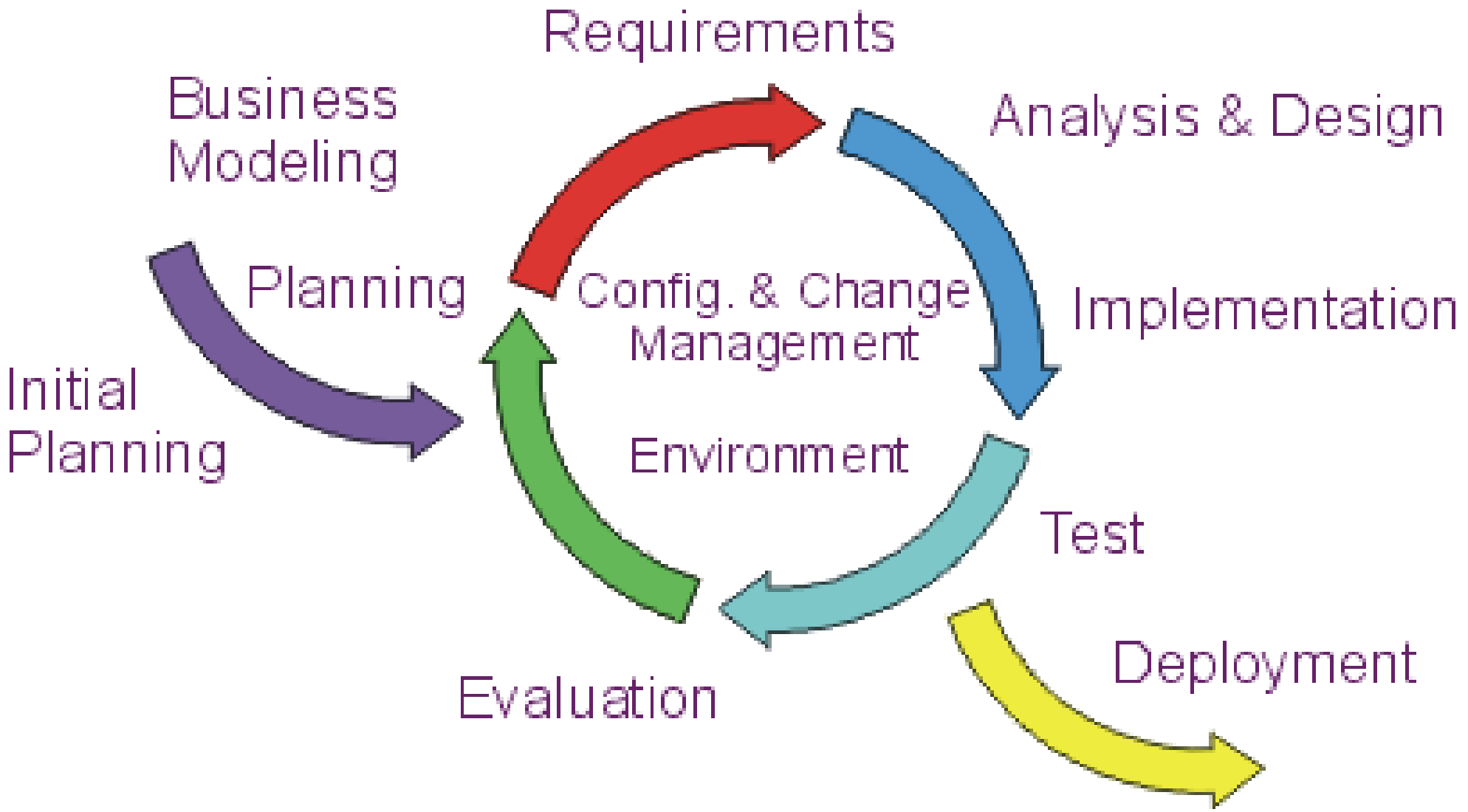
9 DISCIPLINES



Chaque discipline décrit :

- les rôles
- les artefacts
- les activités
- leurs enchaînements

Le processus d'ingénierie logiciel



INCEPTION

- Comprendre le système à construire
- Identifier la fonctionnalité essentielle du système
- Déterminer au moins une architecture possible
- Comprendre les coûts, le calendrier et les risques associés au projet
- Décider du processus à appliquer et des outils à utiliser

ELABORATION

- Comprendre en détail les exigences
- Concevoir, implémenter, valider l'architecture et en établir une version de référence
- Traiter les risques majeurs et estimer plus exactement les délais et le budget
- Affiner le plan de développement et mettre en place l'environnement de développement

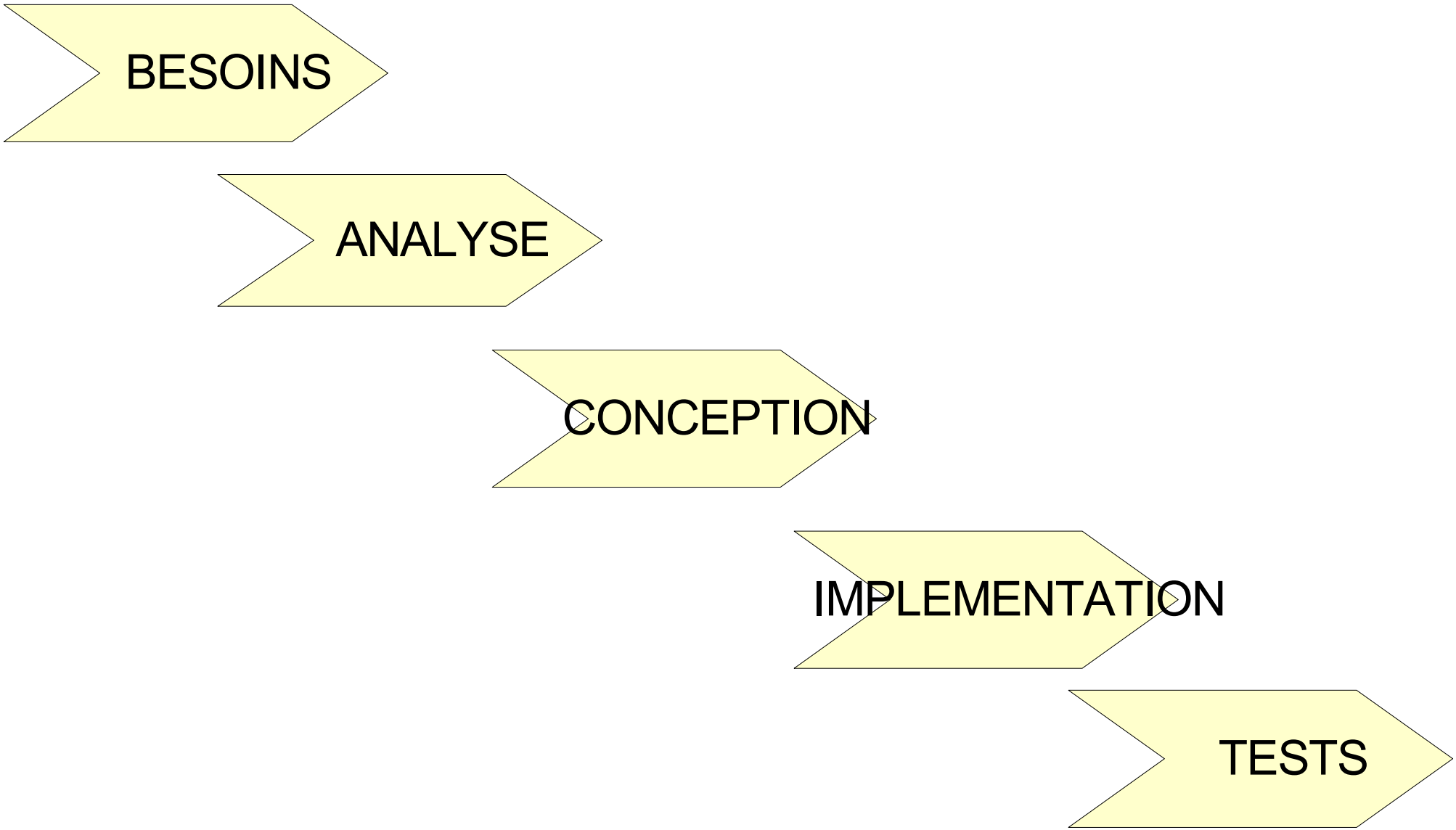
CONSTRUCTION

- Minimiser les coûts de développement et obtenir un certain degré de parallélisme
- Développer de façon itérative un produit complet prêt à la transition vers la communauté des utilisateurs

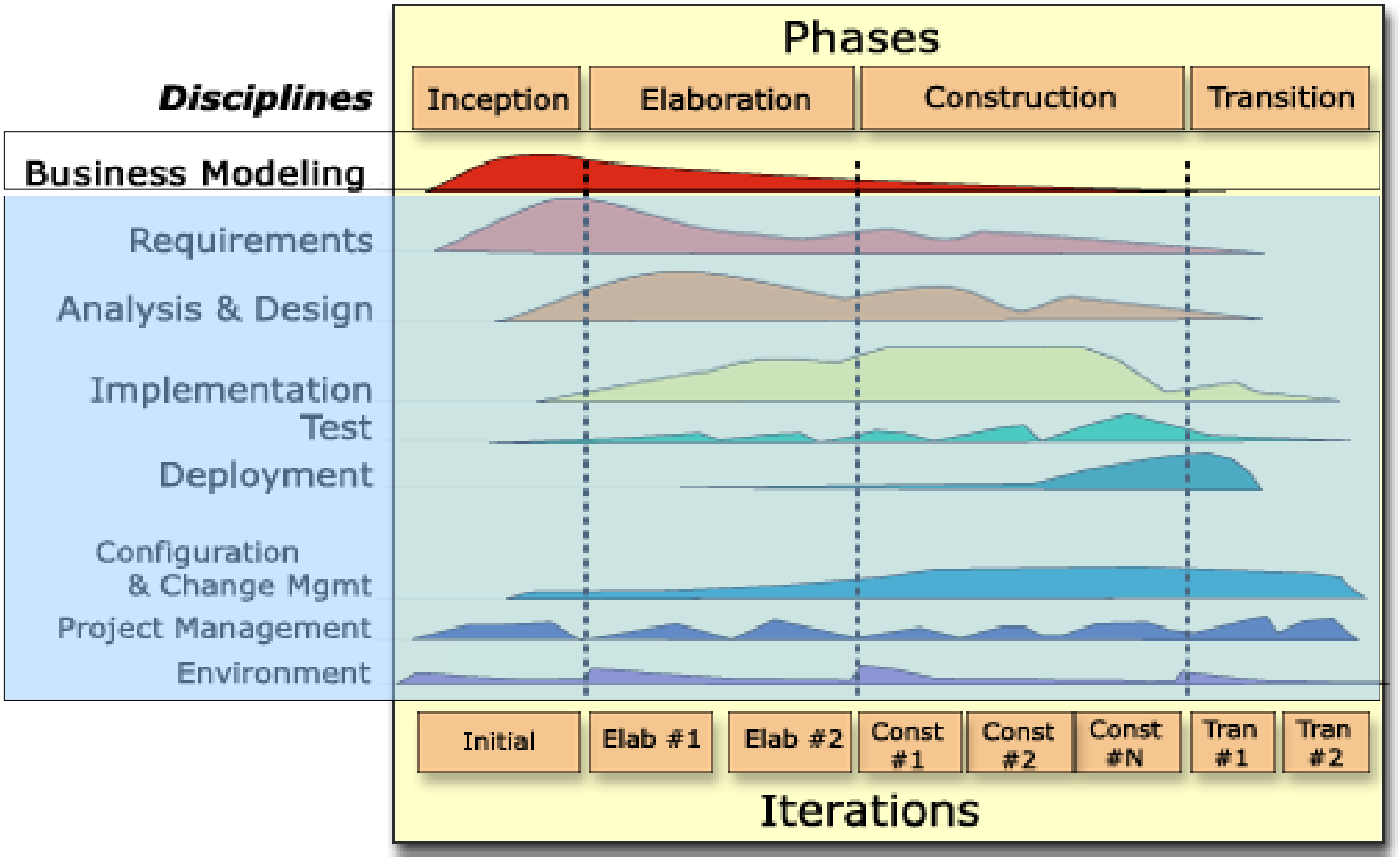
TRANSITION

- Exécuter les tests bêta
- Former les utilisateurs
- Préparer le site déploiements et convertir les bases de données opérationnelles
- Préparer le lancement
- Obtenir l'accord des intervenants
- Améliorer les performances futures

L'ITERATION GENERIQUE



BUSINESS MODELING

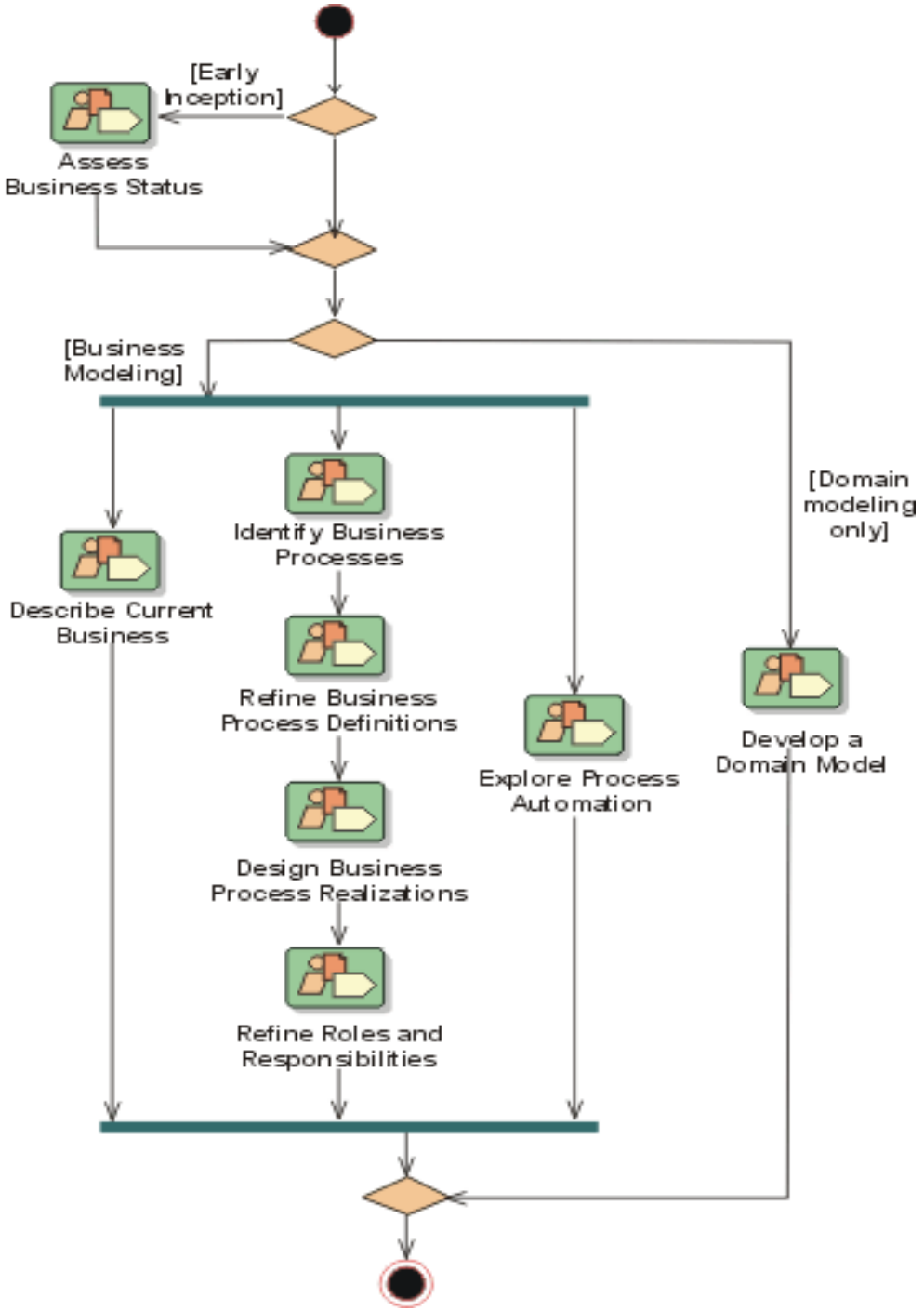


BUSINESS MODELING

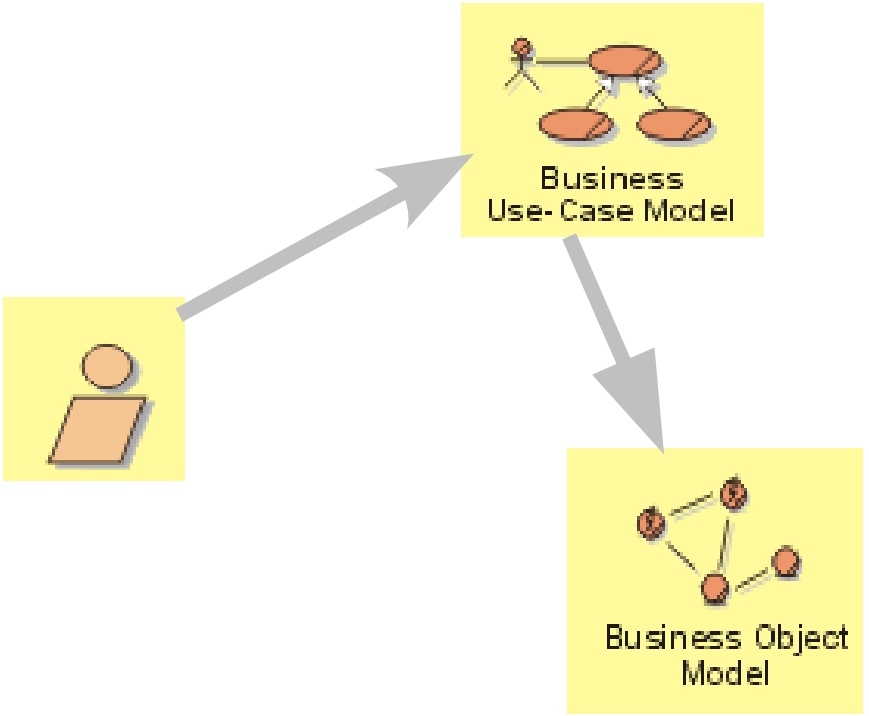
Modéliser les processus métier
- Diagramme de cas d'utilisation métier
- Diagrammes dynamiques

Modéliser les objets métier
- Diagramme de classes métiers
- Diagrammes dynamiques

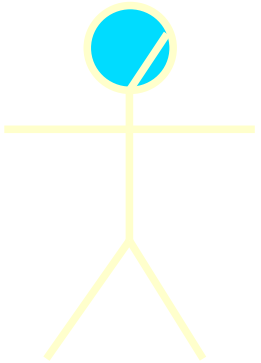
BUSINESS MODELING



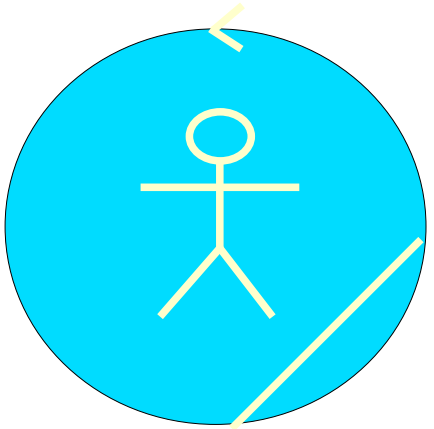
Les principaux artefacts du business modeling



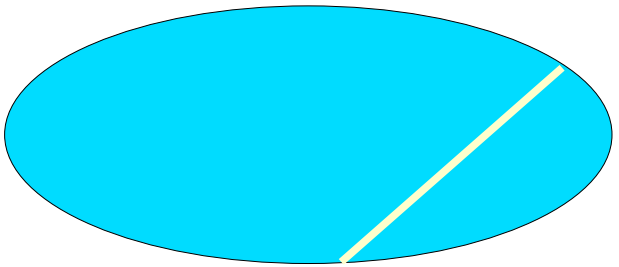
Les stéréotypes de Jacobson



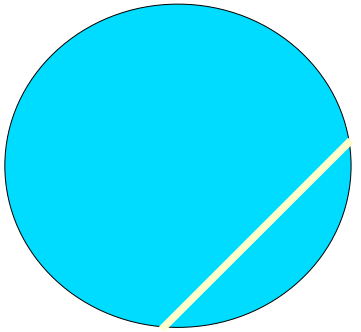
Acteur métier



Travailleur métier



Cas d'utilisation métier



Entité métier

Les concepts de la modélisation métier

- Acteur métier :
  - Personne ou système externe à l'entreprise qui l'utilise ou est utilisée par elle.
- Cas d'utilisation métier :
  - Ensemble d'activités initié par un acteur métier et exécuté par l'entreprise.

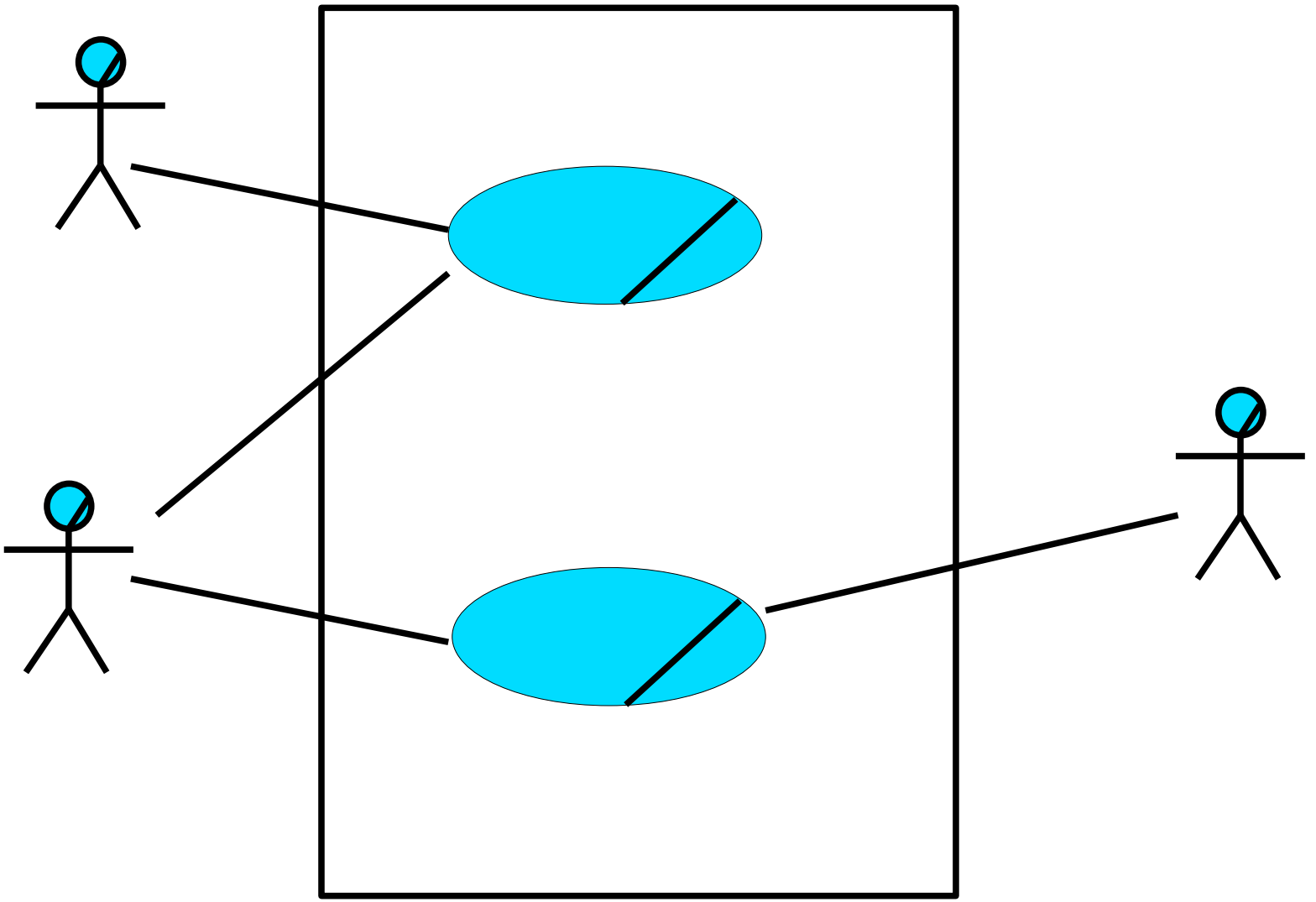
Les concepts de la modélisation métier

- Travailleur métier :
  - Personne appartenant à l'entreprise dont le travail aide à la réalisation d'un cas d'utilisation métier
- Entité métier :
  - Élément utilisé par un travailleur métier pour réalisé un cas d'utilisation métier

Modèle de cas d'utilisation métier

- Il s'agit d'une vue externe de l'entreprise, le point de vue de l'acteur métier.
- Il faut d'abord identifier les acteurs métiers.
- Et pour chaque acteur determiner les cas d'utilisation métier

Diagramme de cas d'utilisation métier



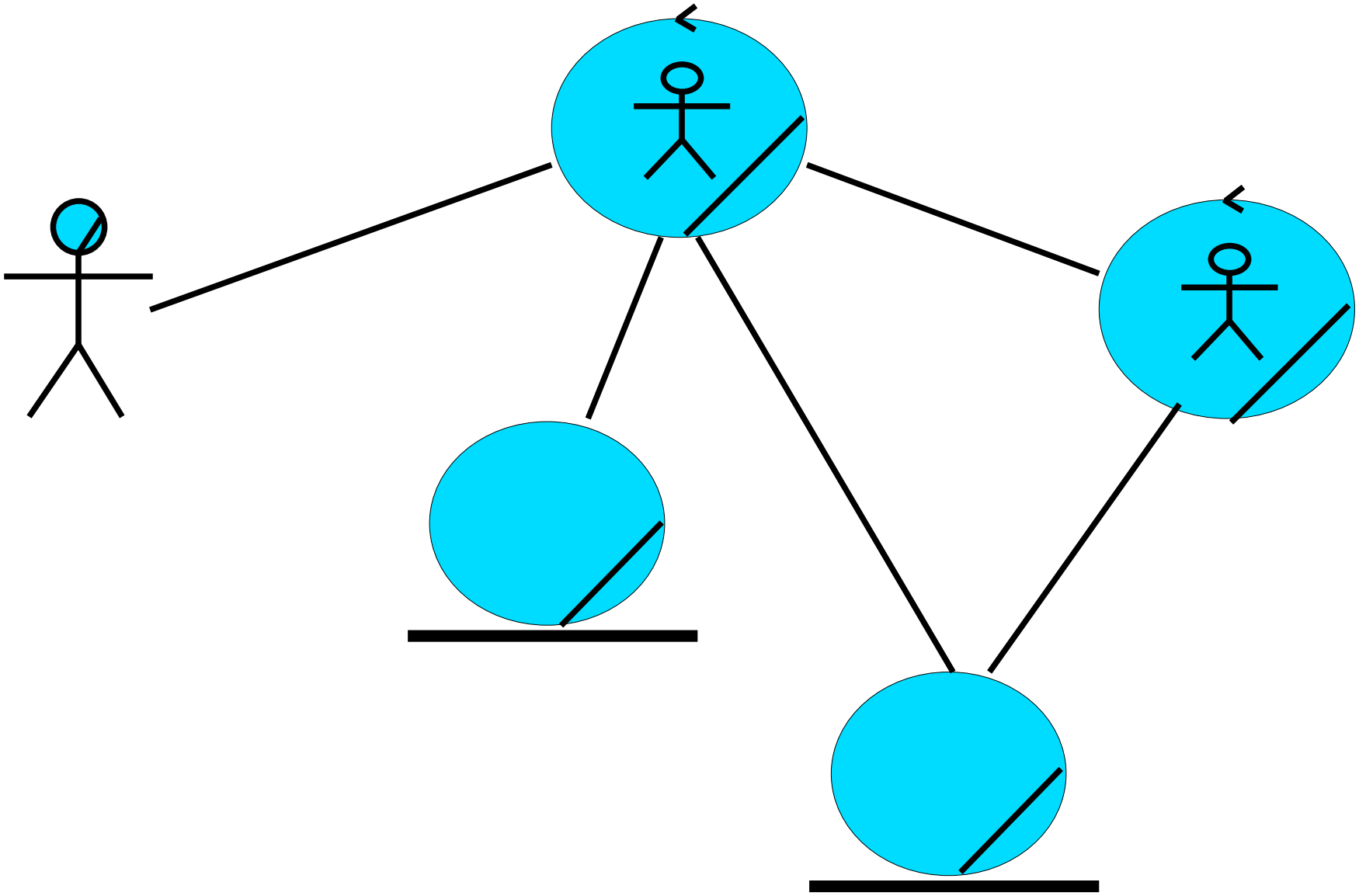
Les diagrammes dynamiques

- Les cas d'utilisation métier sont accompagnés d'un ou de plusieurs diagrammes dynamiques.
- Diagrammes d'activités, de séquence, de communication.
- Il faut s'attacher à respecter le point de vue du ou des acteurs externes

Le modèle objet métier

- Il s'agit d'un diagramme de classe présentant les travailleurs métiers, les acteurs, les entités métiers et leurs relations dans l'accomplissement d'un cas d'utilisation métier.
- Ce diagramme statique est complété de diagrammes d'activités et de séquences.

Le diagramme d'objets métiers

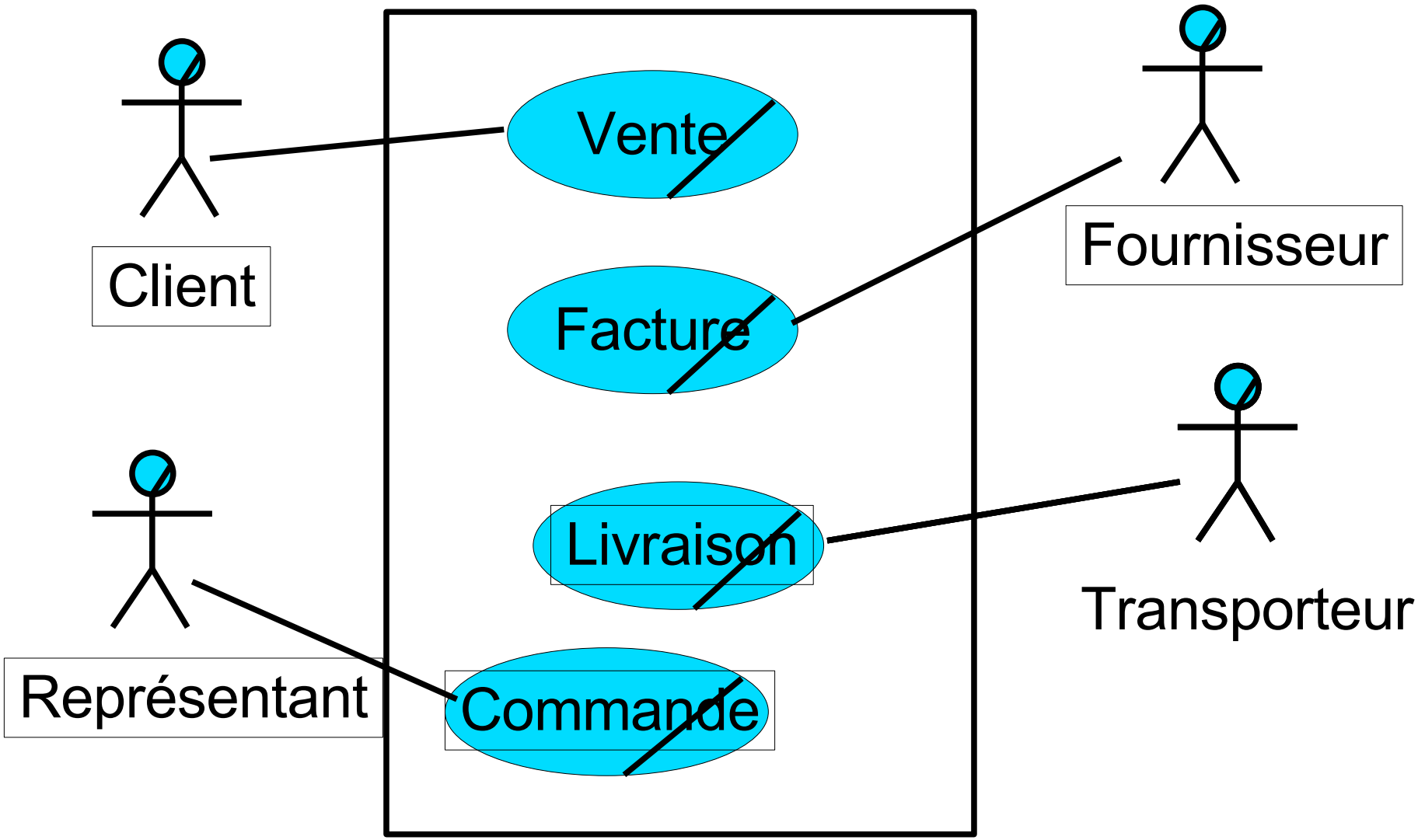


Business modeling : Exercice

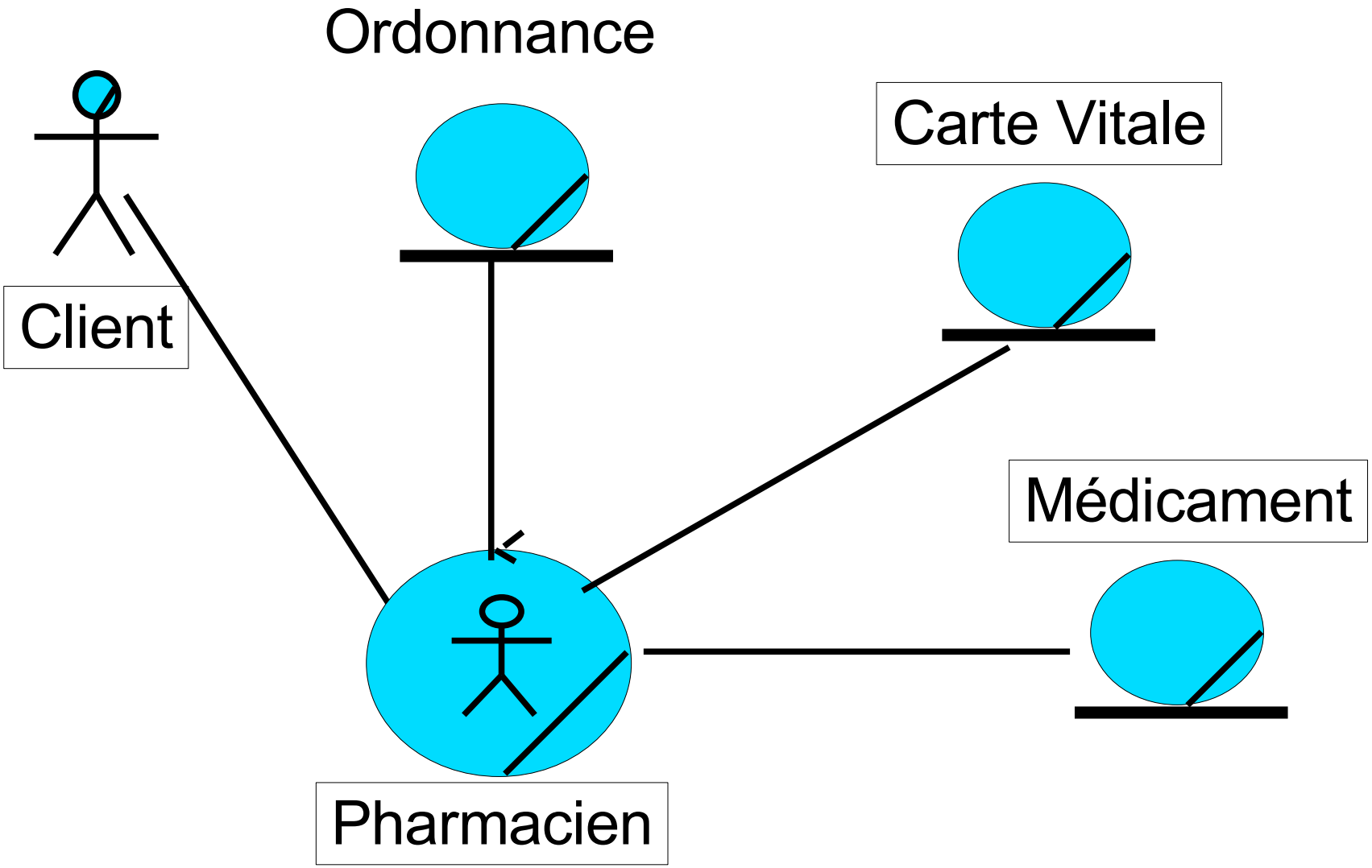
Modélisation d'une pharmacie

- La pharmacie fait uniquement de l'achat/revente à ses clients.
- Elle est sollicitée par des représentants pour différents produits.
- Elle passe commande auprès de ses fournisseurs pour des médicaments qui peuvent être livrés rapidement.

Business modeling : Exercice



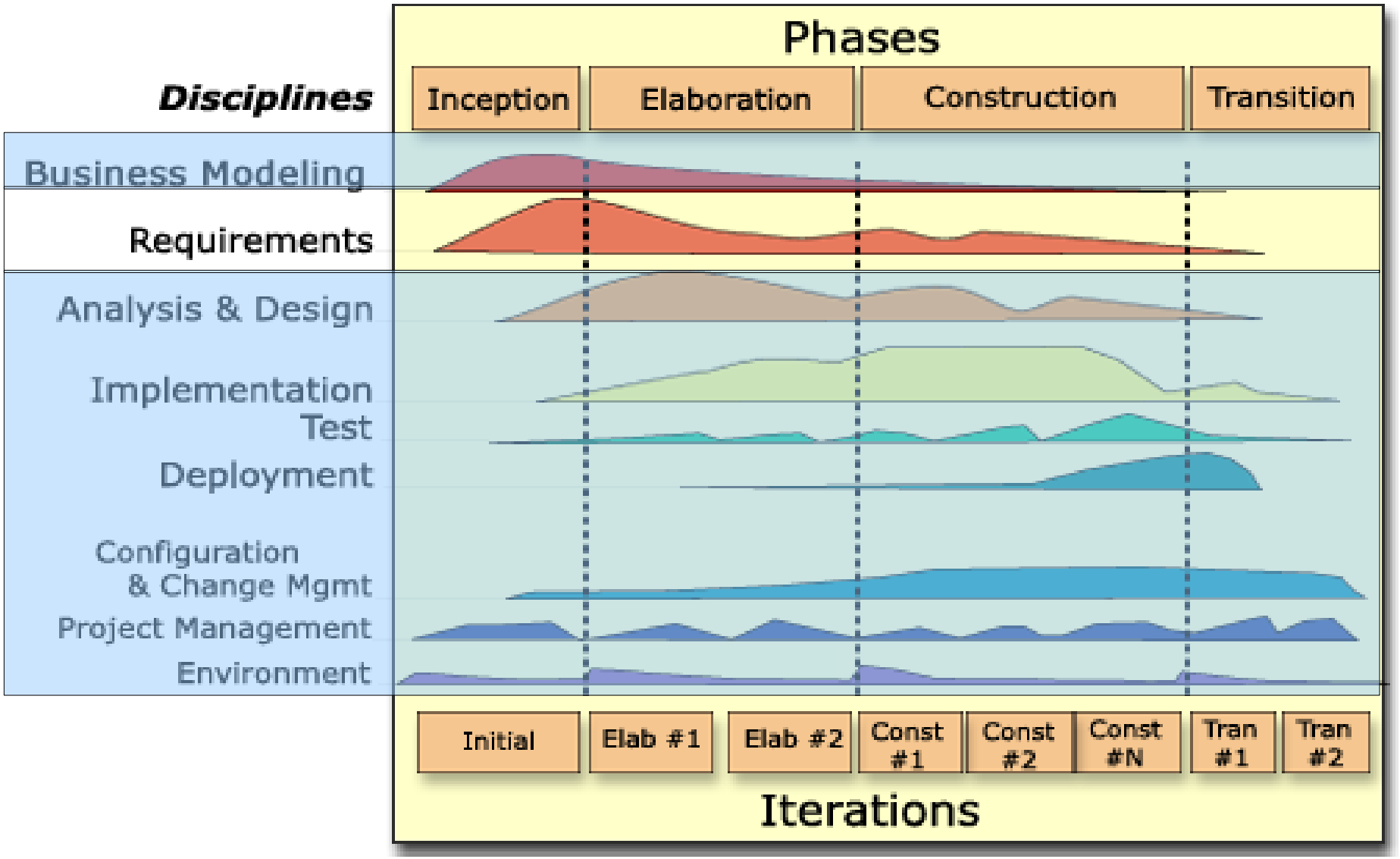
Business modeling : Exercice



Réalisation du cas d'utilisation métier : Achat

EXERCICE

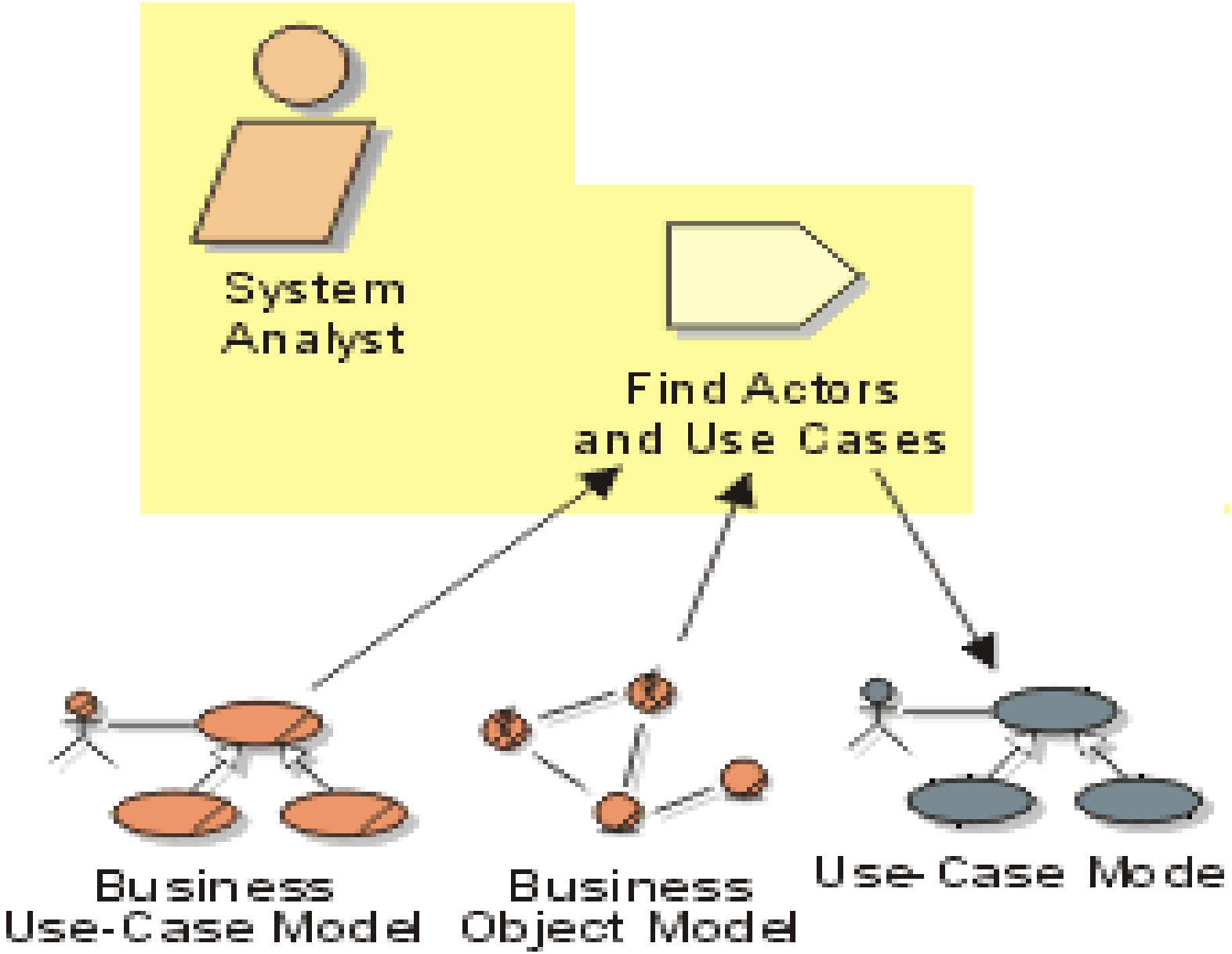
REQUIREMENTS



REQUIREMENTS : Les objectifs

- Exprimer les fonctionnalités attendues sous forme de cas d'utilisation.
- Le recours aux cas d'utilisation oblige l'analyste à réfléchir aux acteurs et leurs besoins.
- D'autre part les cas d'utilisation servent au pilotage du travail de développement

REQUIREMENTS



Modèle des cas d'utilisation

- Permet de parvenir à un accord sur les services que le système doit apporté.
- Il faut identifier l'ensemble des acteurs.
- Chaque usage que les acteurs font du système est représenté par un cas d'utilisation.
- Une description de cas d'utilisation peut être accompagné de diagrammes d'états-transition, d'activités, de collaboration, de séquence.

Description des cas d'utilisation

- Nom
- Description succincte
- Scénario principal
- Scénarios alternatifs
- Pré conditions
- Post conditions

Description de l'architecture

- La vue architecturale doit inclure les fonctionnalités stratégiques.
- Permet de déterminer l'ordre de priorité de conception des cas d'utilisation.

Le glossaire

- Il complète le modèle métier, mais il est ciblé sur le modèle a construire.
- L 'expression des besoins doit être exprimer avec le vocabulaire de l'utilisateur.

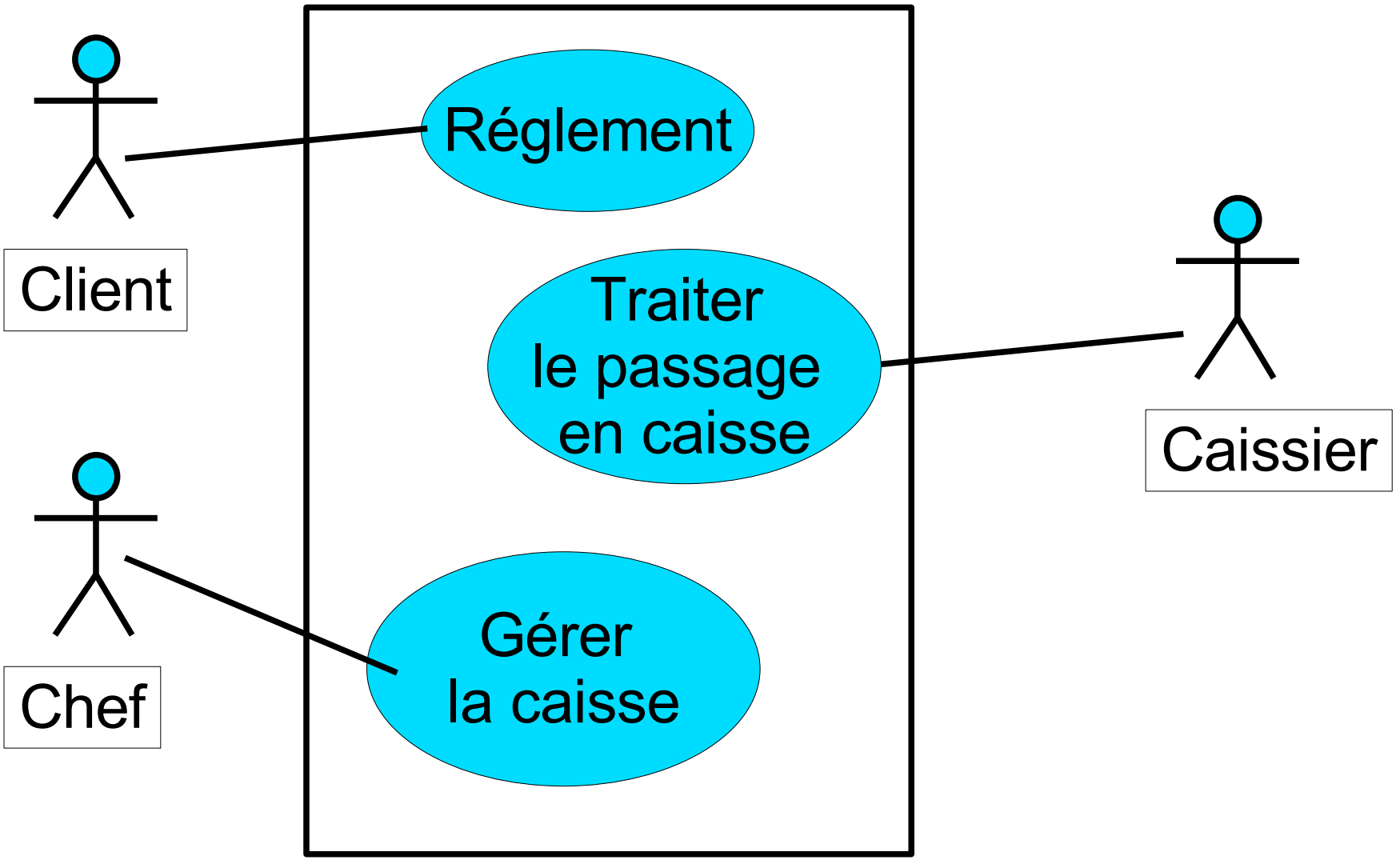
Les prototypes d'interface

- Ils contribuent à l'expression des besoins.
- Et permettent de mieux appréhender les cas d'utilisation.

REQUIREMENTS : Exercice

- Passage en caisse d'un client
- Détermination des acteurs
- Pour chacun des acteurs déterminer les services rendus par le système

REQUIREMENTS : Exercice



REQUIREMENTS : Exercice

Nom : Traiter le passage en caisse
Description succincte : Un client arrive en caisse avec des articles qu'il souhaite acheter.

Le caissier enregistre les achats et le paiement. A la fin de l'opération le client part avec les articles.

REQUIREMENTS : Exercice

Scénario principal :

Le caissier enregistre chaque article et la quantité

La caisse détermine le prix et l'affiche

Quand il n'y a plus d'articles le caissier le signale à la caisse

La caisse calcule et affiche le montant total

...

REQUIREMENTS : Exercice

Scénarios alternatifs :

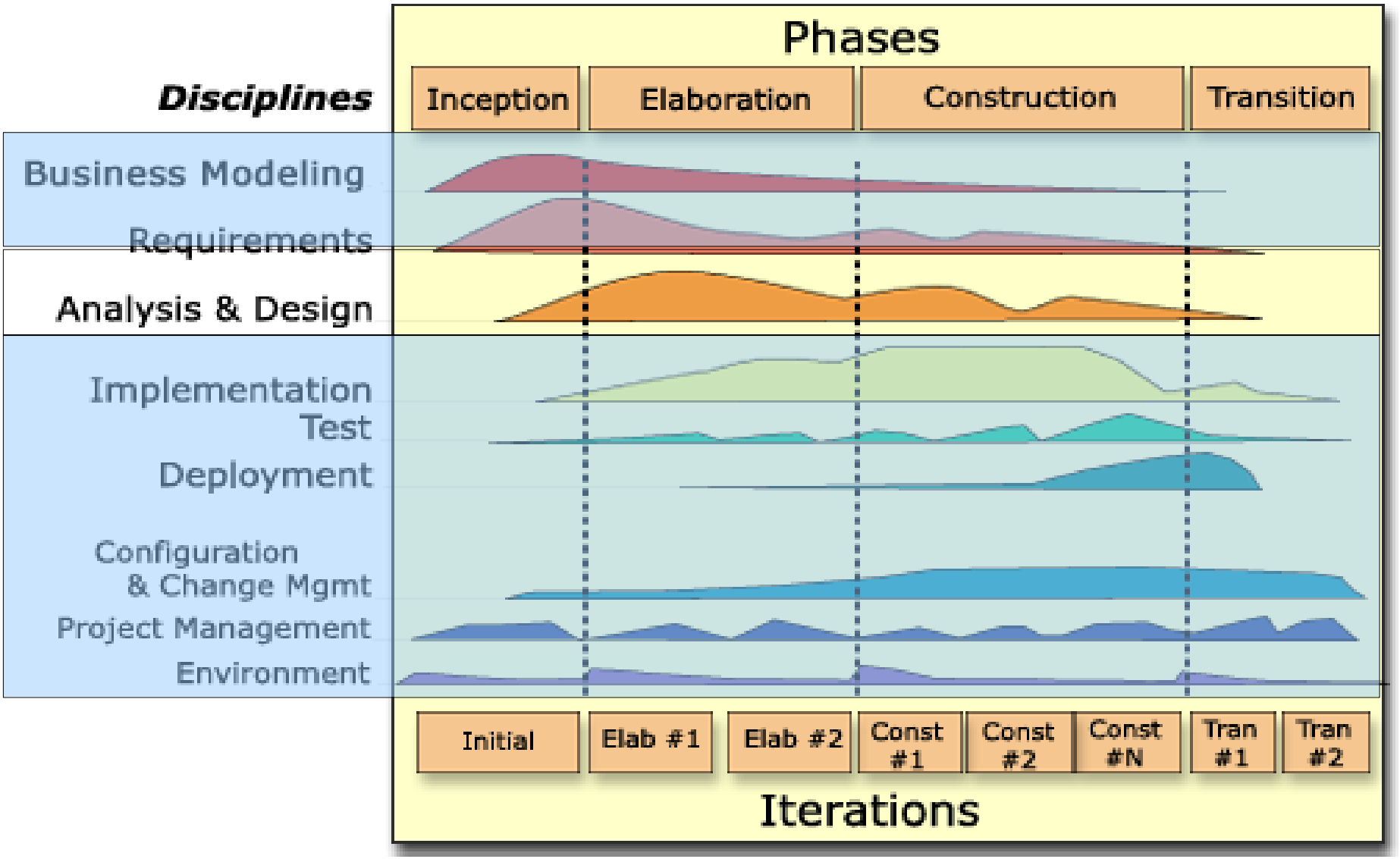
- L'article n'existe pas en stock
- Le client n'a aucun moyen de paiement

Préconditions :

- La caisse est initialisée
- Un caissier est connecté

EXERCICE

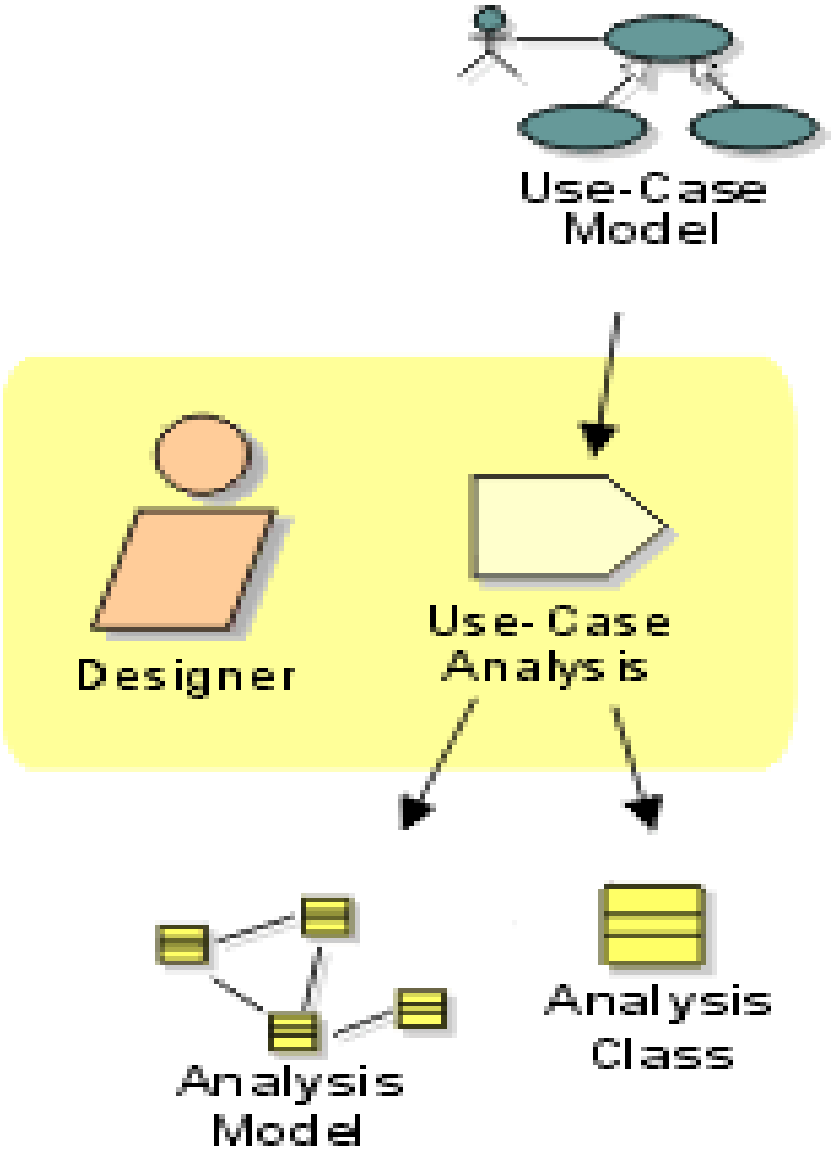
ANALYSIS


Phases
Disciplines
Inception
Elaboration
Construction
Transition
Business Modeling
Requirements
Analysis & Design
Implementation
Test
Deployment
Configuration
& Change Mgmt
Project Management
Environment
Initial
Elab #1
Elab #2
Const #1
Const #2
Const #N
Tran #1
Tran #2
Iterations

ANALYSIS : Les objectifs

- Cette discipline se consacre à l'analyse des besoins définis et décrits dans la discipline Requirements.
- Les cas d'utilisation seront exprimés alors dans le langage des développeurs.
- Les interférences entre cas d'utilisation seront mentionnées.
- L'artefact important de cette discipline est le Modèle d'analyse.

ANALYSIS : principale activité



LE MODELE D'ANALYSE

- Dans le modèle d'analyse, les cas d'utilisation sont réalisés par des classes d'analyse et les responsabilités qu'elles comportent.
- Une classe d'analyse représente l'abstraction d'une ou plusieurs classes de conception.

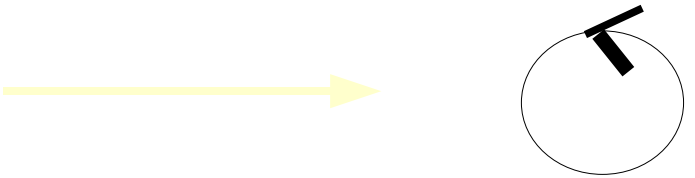
LES CLASSES D'ANALYSE

- Une classe d'analyse présente des responsabilités, qui définissent son comportement.
- Elle peut également comportée des attributs de haut niveau, reconnaissables à partir du domaine.
- Une classe d'analyse appartient toujours à l'un des trois stéréotypes de bases.

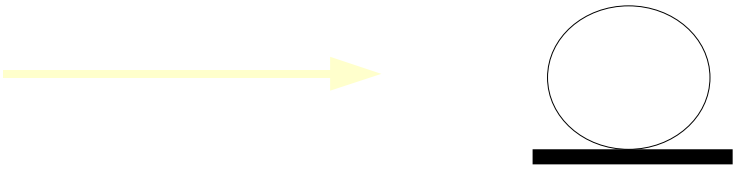
LES CLASSES D'ANALYSE

<<boundary>> Classe A
attributs
responsabilités

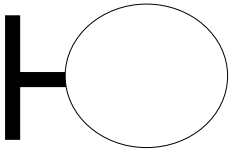
<<control>> Classe B
attributs
responsabilités



<<entity>> Classe C
attributs
responsabilités

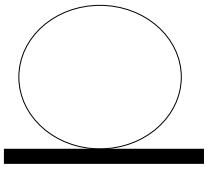


LES CLASSES BOUNDARY



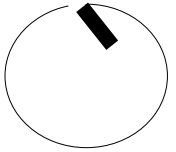
- Modélise l'interaction entre le système et ses acteurs.
- Implique la réception et/ou la présentation d'informations.
- Ces classes doivent demeurer à un niveau conceptuel.
- Elles ne décrivent pas la réalité physique de l'interaction.

LES CLASSES ENTITY



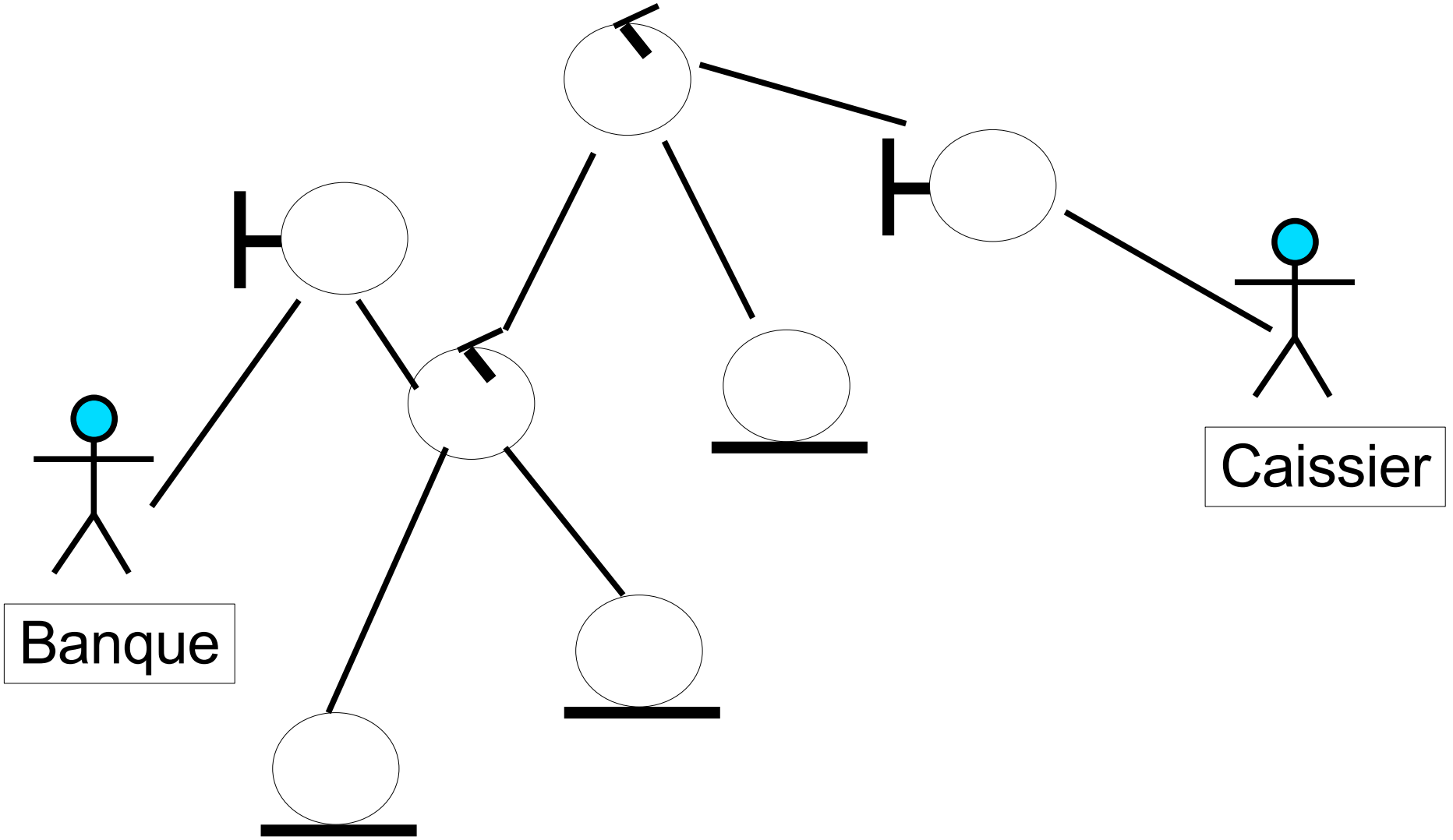
- Sert à modéliser une information de longue durée et de nature persistante.
- Sont en général directement dérivées des classes entités métier du modèle objet métier.
- Fournissent une représentation des informations utiles.

LES CLASSES CONTROL



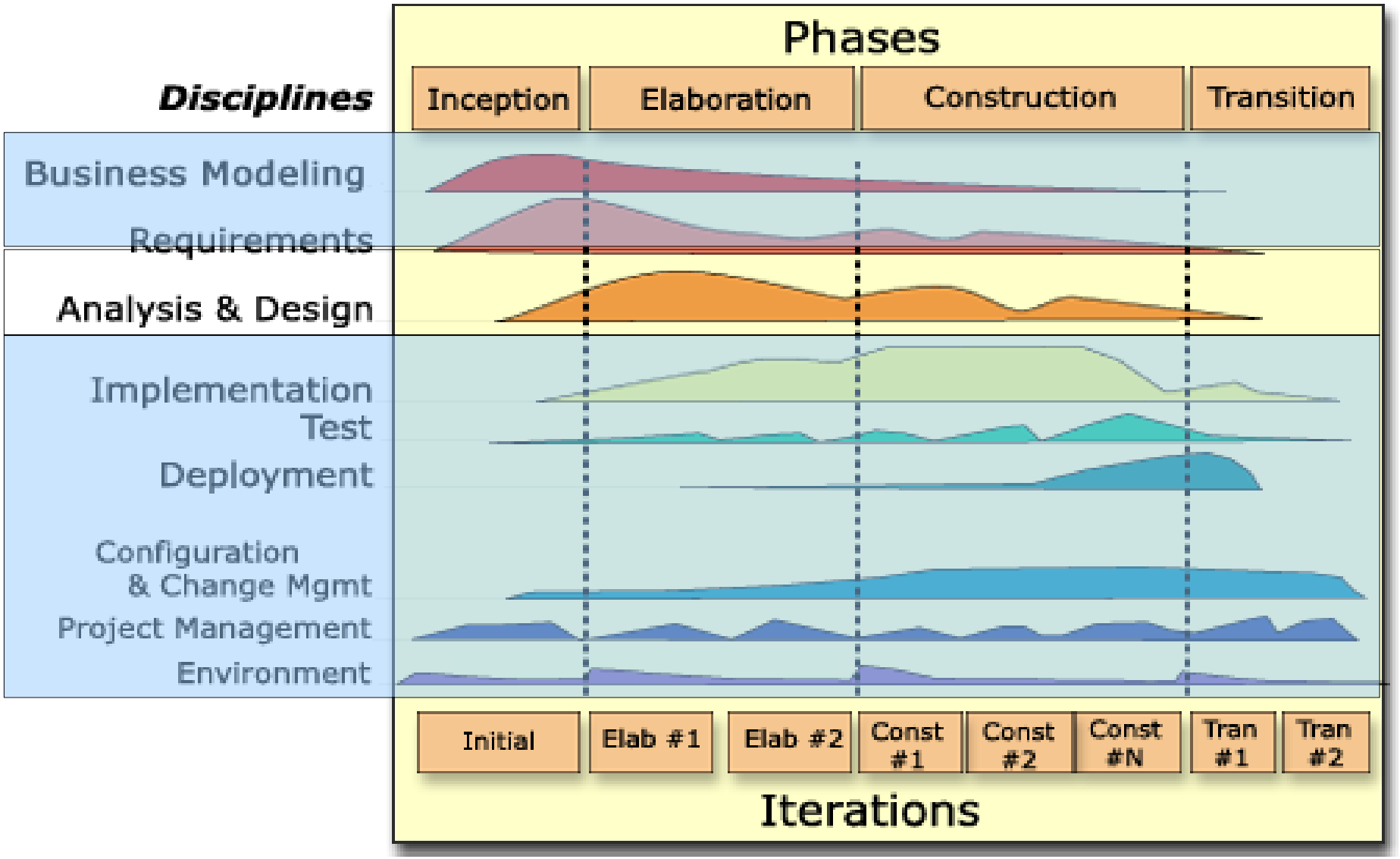
- Elles représentent la coordination et le séquencement.
- Elles servent souvent à encapsuler le déroulement d'un cas d'utilisation

EXEMPLE



EXERCICE

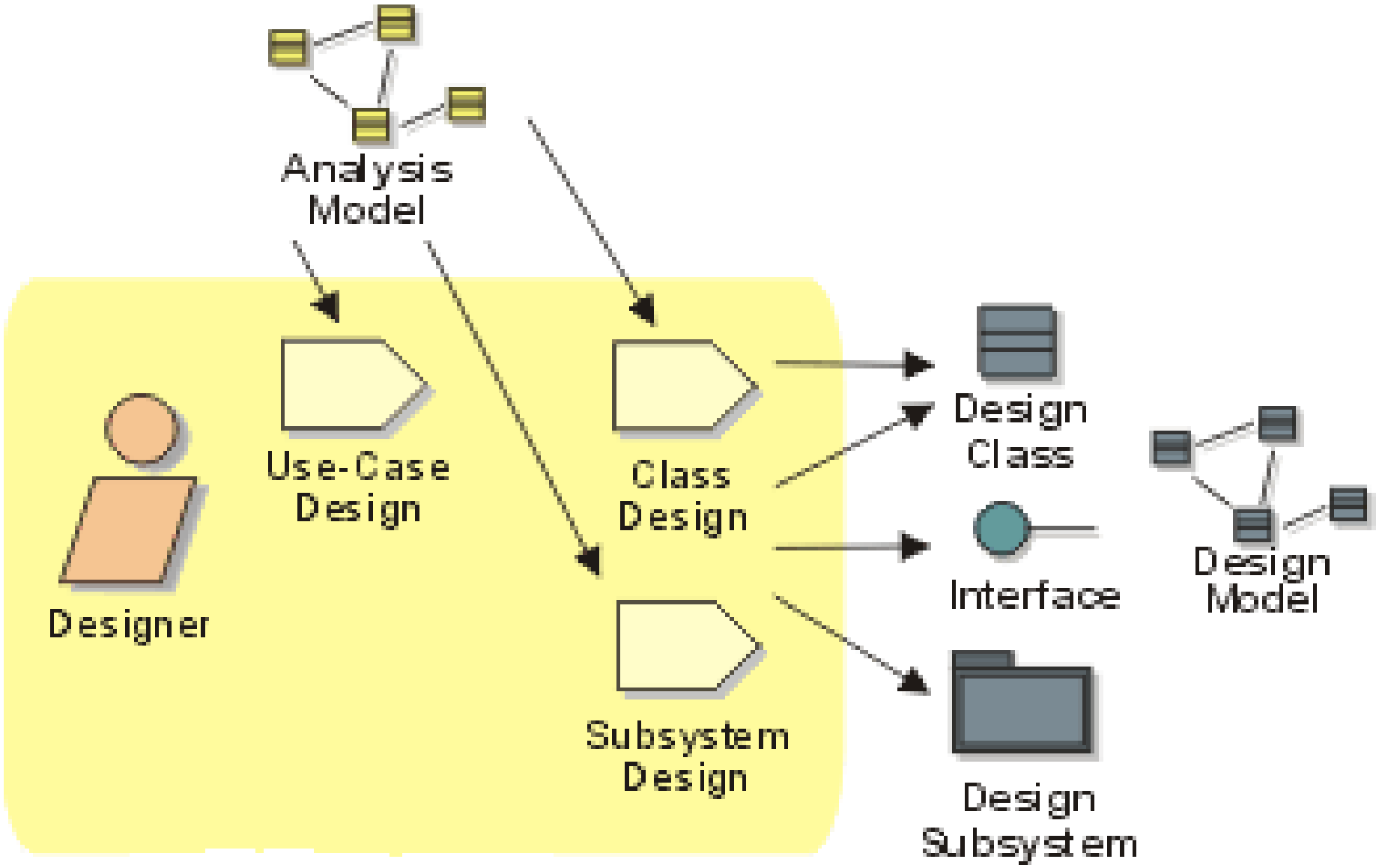
DESIGN


Phases
Disciplines
Inception
Elaboration
Construction
Transition
Business Modeling
Requirements
Analysis & Design
Implementation
Test
Deployment
Configuration
& Change Mgmt
Project Management
Environment
Initial
Elab #1
Elab #2
Const #1
Const #2
Const #N
Tran #1
Tran #2
Iterations

DESIGN : Les objectifs

- Faire le lien entre les classes d'analyse et le langage de programmation.
- Formuler les exigences de façon à ce qu'elles soient implémentées
- Décomposer le travail d'implémentation
- Créer une abstraction du système

DESIGN : Principale activité



DESIGN : Les principaux artefacts

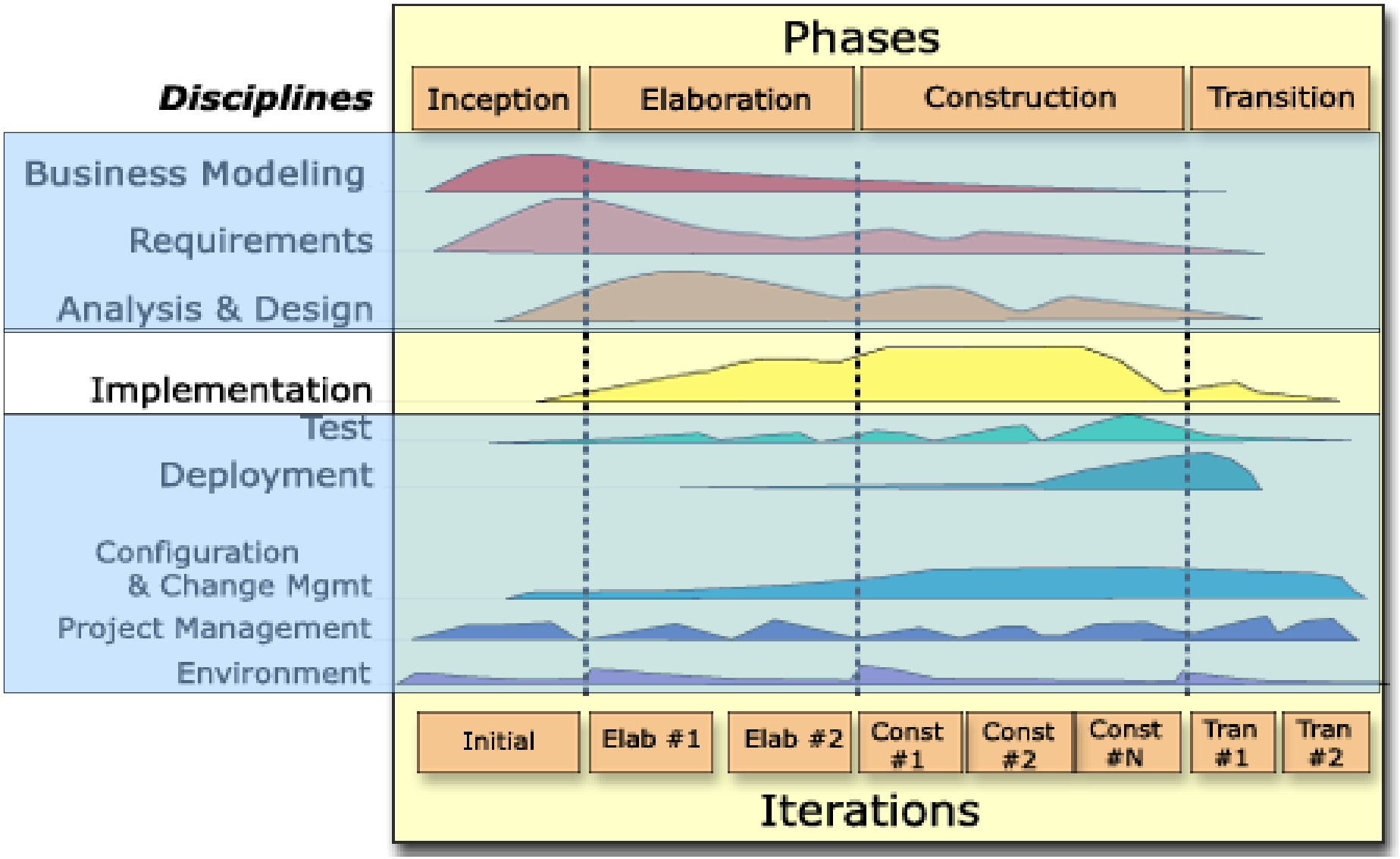
- Les diagrammes de classes
- Les diagrammes d'interaction
- Les diagrammes d'activités
- Les diagrammes d'états
- ...(Cf UML)

LE DIAGRAMME DE CLASSES

- Les classes conçues ne sont plus génériques mais spécifiques à une implémentation
- Les classes de conception sont plus formelles et orientées vers le langage de développement choisi

EXERCICE

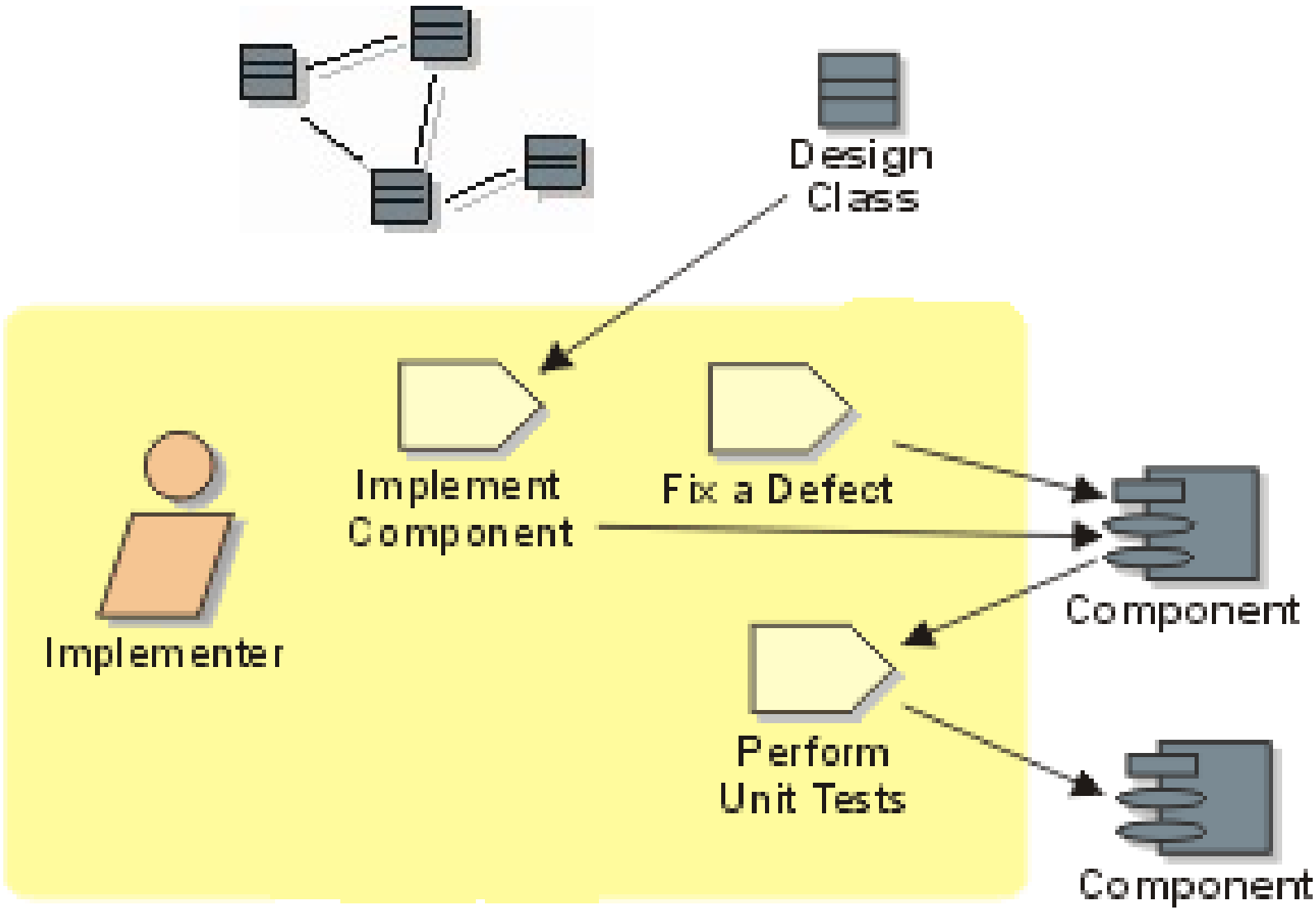
IMPLEMENTATION



IMPLEMENTATION : Les objectifs

- Implémenter les classes de conception
- Tester chaque composant avant de faire l'intégration
- Planifier l'intégration

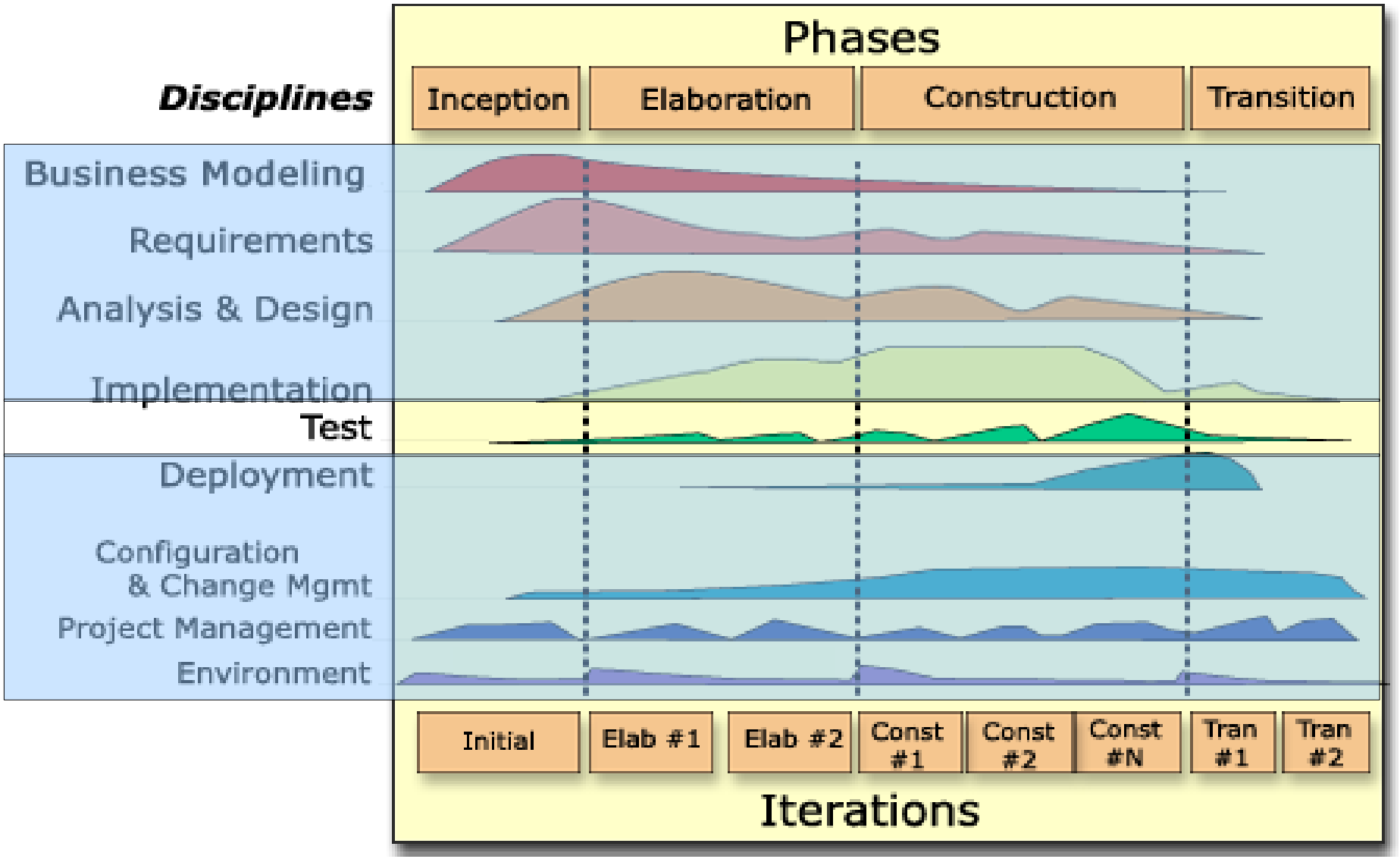
IMPLEMENTATION : Principale activité



IMPLEMENTATION : Les principaux artefacts

- Les exécutables
- Le fichiers sources ou de données
- Les bibliothèques statiques ou dynamiques
- Les tables

TEST


Phases
Disciplines
Inception
Elaboration
Construction
Transition
Business Modeling
Requirements
Analysis & Design
Implementation
Test
Deployment
Configuration
& Change Mgmt
Project Management
Environment
Initial
Elab #1
Elab #2
Const #1
Const #2
Const #N
Tran #1
Tran #2
Iterations

TEST : Les objectifs

- L'activité de test s'attache à vérifier les résultats de l'implémentation
- Tests d'intégration à chaque construction du système
- Les tests doivent être planifiés, conçus et implémentés de façon à être le plus possible automatisés

TEST : Les principaux artefacts

- Cas de test
- Procédure de test
- Composant de test